Estratégia Concursos

Aula 02

IFCE (Prof. Ciência da Computação-Metodologia e Técnicas da Comput) Conhecimentos - 2021(Pós-Edital)

Autor:

Diego Carvalho, Equipe Informática e TI, Evandro Dalla Vecchia Pereira , Pedro Henrique Chagas Freitas, Thiago Rodrigues

Cavalcanti, Raphael Henrique Lacerda

Aula 02

20 de Setembro de 2021

Sumário

APRESENTAÇÃO DA AULA

Pessoal, o tema da nossa aula é Estrutura de dados. **Galera, esse é um assunto um pouco assustador para quem nunca programou, mas não fiquem com medo dele, meu trabalho aqui é fazer com que o assunto seja didático, leve e simples.**

Professor Pedro Freitas
www.instagram.com/pedro.satierf

Galera, todos os tópicos da aula possuem Faixas de Incidência, que indicam se o assunto cai muito ou pouco em prova. *Pedro, se cai pouco para que colocar em aula?* Cair pouco não significa que não cairá justamente na sua prova! A ideia aqui é: se você está com pouco tempo e precisa ver somente aquilo que cai mais, você pode filtrar pelas incidências média, alta e altíssima; se você tem tempo sobrando e quer ver tudo, vejam também as incidências baixas e baixíssimas. *Fechado?*

INCIDÊNCIA EM PROVA: baixíssima
INCIDÊNCIA EM PROVA: baixa
INCIDÊNCIA EM PROVA: média
INCIDÊNCIA EM PROVA: ALTA
INCIDÊNCIA EM PROVA: Altíssima

RECADO DA EQUIPE DE TI DO ESTRATÉGIA

Hoje eu faço parte de uma equipe **SENSACIONAL** de professores! Depois de muita luta conseguimos reunir **um time** de profissionais extremamente QUALIFICADO e sobretudo **COMPROMISSADO** em fazer o melhor pelos alunos. Para tal criamos um conjunto de ações para nos aproximarmos dos alunos, entendermos suas necessidades e evoluirmos nosso material para um patamar ainda mais diferenciado. São 3 as novidades que gostaria de convidá-lo a conhecer:

//estratégia tech ESTRATEGIA CONCURSOS	Nosso podcast alternativo ... livre, descontraído e com dicas rápidas que todo CANETA PRETA raiz deve ouvir. Já temos alguns episódios disponíveis e vários outros serão gravados nas próximas semanas ... acompanhe em: *http://anchor.fm/estrategia-tech*
Telegram a new era of messaging	Nosso grupo do Telegram é um local onde ouvimos os alunos e trocamos ideias com eles. Está crescendo a cada dia. A regra do grupo é: só vale falar sobre concursos. Lá divulgamos nossas aulas ao vivo e falamos sobre os concursos abertos, expectativas de novos concursos, revisões de véspera, e por aí vai... *http://t.me/estrategia_ti*
Instagram	Criamos um perfil no Instagram ... e qual o objetivo? Fazer com que os alunos percam tempo nas redes sociais? Claro que não!! Estamos consolidando diversos posts dos professores! São dicas especiais, um patrimônio que deve ser explorado por todos os concurseiros de TI! *http://instagram.com/estrategiaconcursosti*

APRESENTAÇÃO PESSOAL

Sou Engenheiro de Computação, Especialista em Gestão e Desenvolvimento de Sistemas com Mestrado em Gestão do Conhecimento e Tecnologia da Informação e Doutorando em Engenharia de Software.

Antes de tudo: sou caneta preta! Concurseiro desde 2012, então já temos quase 1 década de peleja, somando 6 aprovações, sendo uma delas como primeiro colocado.

Atualmente, sou Analista de Tecnologia do Ministério da Economia, lotado na Secretaria de Governo Digital, onde ingressei aos 23 anos de idade. De lá para cá fui estudando duro, trabalhando forte e galgando algumas posições, como: Coordenador Geral de Inovação e Informações Estratégicas, Assessor de Gabinete, Coordenador de Governança, Assessor Técnico da Presidência e atualmente sou em tempo integral: Pai, Esposo e Filho, além de Professor é claro.

Minha estrada Profissional:

Fui aprovado na 1º e 2º fase para o cargo de Subcontrolador de Governo Aberto pelo Estado de Minas Gerais.

Ganhei em 2018 o prêmio de melhor projeto de inovação na Administração Pública na Semana da Inovação do Governo Federal entregue pelo Ministro Esteves Colnago.

Conduzi a mudança de paradigma de desenvolvimento do Ministério da Agricultura para Arquitetura orientada a serviço (SOA).

Criei a matriz de cursos e competências de transformação digital da ENAP, em espelhamento a matriz de Competências para transformação digital do Reino Unido.

Assessorei vários projetos no Ministério da Economia, Ministério da Cidadania e Presidência da República.

Colaborei no batimento e descoberta de dados no Cadastro Único e no ENEM, que fizeram com que o Governo Federal, encontra-se os prodígios brasileiros em situação de fragilidade social.

Vida de Professor:

Quanto à atividade de professor, já lecionei disciplinas de Tecnologia da Informação, Engenharias e Raciocínio Lógico para: concursos, graduação e pós-graduação. Escrevo também para a UOL Tech sobre: Deep Learning e Programação em Python (Melhor linguagem para vocês aprenderem, **#FICADICA**). E hoje faço parte do Estratégia e trabalho dia e noite para ajudar nossos alunos a alcançar o sonho da aprovação.

Quaisquer dúvidas, sugestões ou reflexões, ai em baixo tem o meu e-mail. Terei o prazer em orientá-los da melhor forma possível e de somar com vocês, temos também vários outros cursos:

E-mail do Professor	professorpedrofreitas@gmail.com
Cursos Estratégias	*https://www.estrategiaconcursos.com.br/cursosPorProfessor/pedro-henrique-chagas-freitas-4000/*

ESTRUTURA DE DADOS

1 – Conceitos gerais

INCIDÊNCIA EM PROVA: baixa

Pessoal, um programa pode ser visto como uma especificação formal da solução de um problema. **Wirth expressa esse conceito por meio de uma equação**:

PROGRAMA = ALGORITMO + ESTRUTURA DE DADOS

Nosso foco aqui é em Estruturas de Dados! **Na evolução do mundo computacional, um fator extremamente importante trata da forma de armazenar informações**. De nada adianta o enorme desenvolvimento de hardware e software se a forma de armazenamento e tratamento de dados não evoluir harmonicamente. E é por isso que as estruturas de dados são tão fundamentais.

As estruturas de dados, na maioria dos casos, baseiam-se nos tipos de armazenamento vistos dia a dia, i.e., nada mais são do que a transformação de uma forma de armazenamento já conhecida e utilizada no mundo real adaptada para o mundo computacional. **Por isso, cada tipo de estrutura de dados possui vantagens e desvantagens e cada uma tem sua área de atuação otimizada**.

Bem, não vou enrolar muito explicando o que é uma Estrutura de Dados! A melhor forma de saber é vendo exemplos. **Antes disso, eu gostaria de falar sobre um conceito importante: Dados Homogêneos e Heterogêneos.** Os primeiros são aqueles que possuem só um tipo básico de dados (Ex: Inteiros); os segundos são aqueles que possuem mais de um tipo básico de dados (Ex: Inteiros + Caracteres). Os tipos básicos de dados também são chamados de tipos primitivos.

Entenderam? **Existem estruturas de dados que tratam de dados homogêneos, i.e., todos os dados são apenas de um tipo básico, tais como Vetores!** Ora, em um vetor, todos os elementos são do mesmo tipo. Existem estruturas de dados que tratam de dados heterogêneos, i.e., os dados são de tipos básicos diferentes, tais como Listas! Ora, em uma lista, todos os elementos são, em geral, de tipos básicos diferentes.

Além dessa classificação, existe outra também importante: Estruturas Lineares e Estruturas Não-Lineares. As Estruturas Lineares são aquelas em que cada elemento pode ter um único predecessor (exceto o primeiro elemento) e um único sucessor (exceto o último elemento). Como exemplo, podemos citar Listas, Pilhas, Filas, Arranjos, entre outros.

Já as Estruturas Não-Lineares são aquelas em que cada elemento pode ter mais de um predecessor e/ou mais de um sucessor. **Como exemplo, podemos citar Árvores, Grafos e Tabelas de Dispersão. Essa é uma classificação muito importante e muito simples de entender.** Pessoal, vocês perceberão que esse assunto é cobrado de maneira superficial na maioria das questões, mas algumas são nível doutorado!

Por fim, vamos falar sobre Tipos Abstratos de Dados (TAD). Podemos defini-lo como um modelo matemático **(v,o)**, em que **v** é um conjunto de valores e **o** é um conjunto de operações que podem ser realizadas sobre valores. **Eu sei, essa definição é horrível de entender! Para compreender esse conceito, basta lembrar de abstração, i.e., apresentar interfaces e esconder detalhes.**

Os Tipos Abstratos de Dados são simplesmente um modelo para um certo tipo de estrutura de dados. *Como assim, professor?* Quando eu falo em pilha, eu estou falando de um tipo abstrato de dados que tem duas operações com comportamentos bem definidos e conhecidos: *push* (para inserir elementos na pilha); e *pop* (para retirar elementos da pilha).

E a implementação dessas operações? Isso não importa! **Aliás, não importa implementação nem paradigma nem linguagem de programação.** Não importa se a pilha é implementada com um paradigma orientado a objetos ou com um paradigma estruturado; não importa se a pilha é implementada em Java ou Pascal; não importa como é a implementação interna – isso serve para outras estruturas[1].

Podemos concluir, portanto, que um tipo abstrato de dados contém um modelo que contém valores e operações associadas, de forma que essas operações sejam precisamente independentes de uma

[1] Filas, Pilhas, Árvores, Deques, entre outros.

implementação particular. **Em geral, um TAD é especificado por meio de uma especificação algébrica que, em geral, contém três partes: Especificação Sintática, Semântica e de Restrições**.

A Especificação Sintática define o nome do tipo, suas operações e o tipo dos argumentos das operações, definindo a assinatura do TAD. **A Especificação Semântica descreve propriedades e efeitos das operações de forma independente de uma implementação específica**. E a Especificação de Restrições estabelece as condições que devem ser satisfeitas antes e depois da aplicação das operações.

Em outras palavras, o nível semântico trata do comportamento de um tipo abstrato de dados; e o nível sintático trata da apresentação de um tipo abstrato de dados. **Podemos dizer, então, que o TAD encapsula uma estrutura de dados com características semelhantes – podendo ser formado por outros TADs –, e esconde a efetiva implementação dessa estrutura de quem a manipula.**

(CRM-MT - 2021) Uma estrutura de dados pode ser entendida como um padrão de organização de dados que permita ao computador o acesso e utilização destes dados de forma facilitada. A literatura define diversos tipos de estrutura de dados. Cada uma delas possui características próprias, sendo indicadas na solução de problemas específicos.

_____

Comentários: Perfeita a descrição: Uma estrutura de dados pode ser entendida como um padrão de organização de dados que permita ao computador o acesso e utilização destes dados de forma facilitada. A literatura define diversos tipos de estrutura de dados. Cada uma delas possui características próprias, sendo indicadas na solução de problemas específicos.

(CORRETO).

(IF-RR - 2021) Em Ciência da Computação, as Estruturas de Dados definem como os dados podem ser organizados, bem como quais operações podem ser realizadas para manipular esses dados.

_____

Comentários: Exatamente, as estruturas de dados definem como os dados podem ser organizados, bem como quais operações podem ser realizadas para manipular esses dados.

(CORRETO).

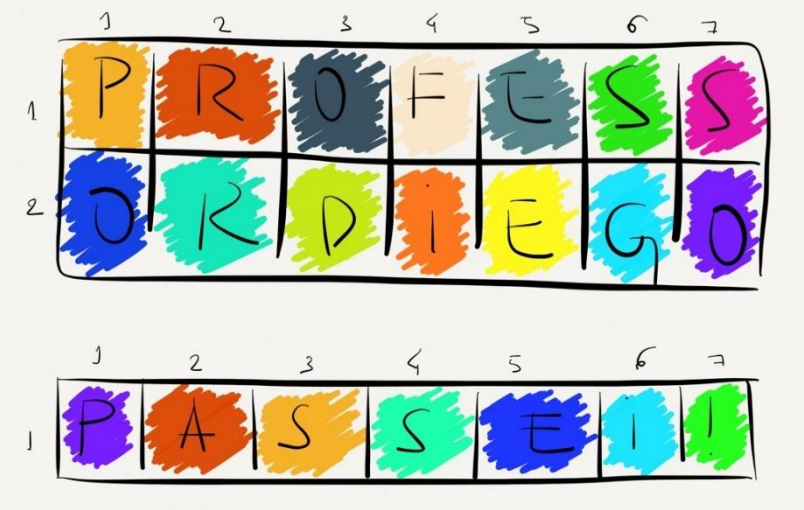
2 – Estruturas de dados: Vetores e Matrizes

INCIDÊNCIA EM PROVA: média

Um Vetor é uma estrutura de dados linear que necessita de somente um índice para que seus elementos sejam indexados. É uma estrutura homogênea, portanto armazena somente uma lista de valores do mesmo tipo. Ele pode ser estático ou dinâmico, com dados armazenados em posições contíguas de memória e permite o acesso direto ou aleatório a seus elementos.

Observem que, diferentemente das listas, filas e pilhas, ele vem praticamente embutido em qualquer linguagem de programação. *E a Matriz, professor?* Não muda muita coisa! Trata-se de um arranjo bidimensional ou multidimensional de alocação estática e sequencial. Ela necessita de um índice para referenciar a linha e outro para referenciar a coluna para que seus elementos sejam endereçados.

Da mesma forma que um vetor, uma matriz também pode ter tamanhos variados, todos os elementos são do mesmo tipo, cada célula contém somente um valor e os tamanhos dos valores são os mesmos. Os elementos ocupam posições contíguas na memória. **A alocação dos elementos pode ser feita colocando os elementos linha-por-linha ou coluna-por-coluna**.



MATRIZ 2X7 E VETOR (7 POSIÇÕES)

3 – Estruturas de dados: Lista encadeada

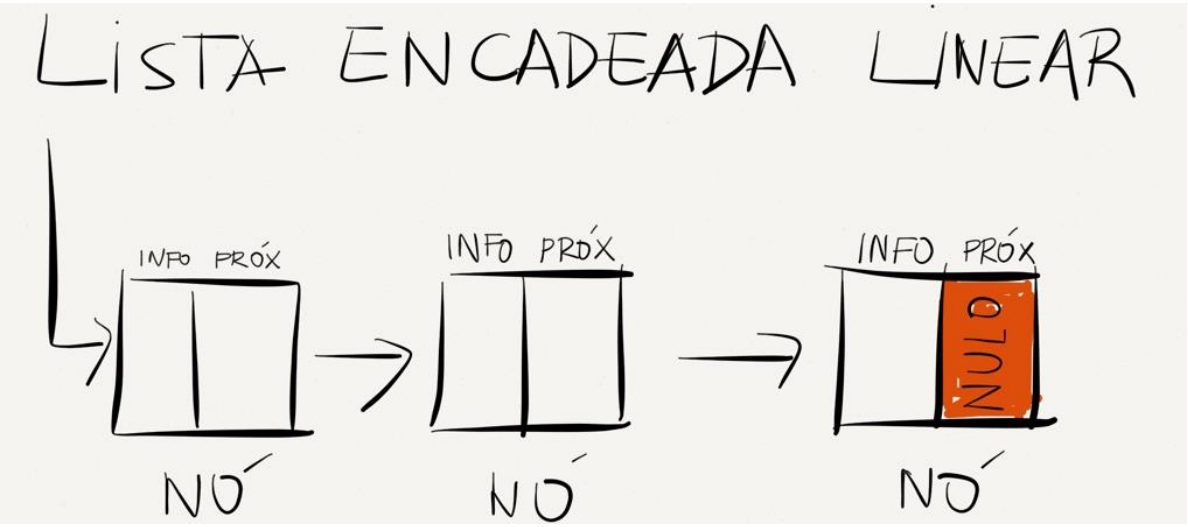
INCIDÊNCIA EM PROVA: média

Também conhecida como *Lista Encadeada Linear*, *Lista Ligada Linear* ou *Linked List*, trata-se de uma estrutura de dados dinâmica formada por uma sequência encadeada de elementos chamados nós, que contêm dois campos: **campo de informação** e **campo de endereço**. O primeiro armazena o real elemento da lista e o segundo contém o endereço do próximo nó da lista.

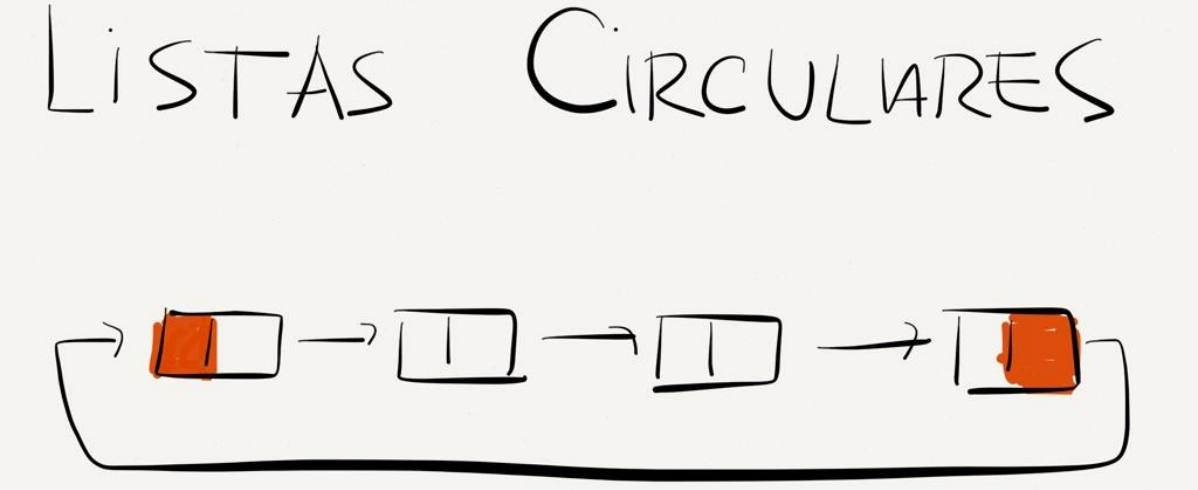
Esse endereço, que é usado para acessar determinado nó, é conhecido como ponteiro. A lista ligada inteira é acessada a partir de um ponteiro externo que aponta para o primeiro nó na lista, i.e., contém o endereço do primeiro nó[2]. Por ponteiro "externo", entendemos aquele que não está incluído dentro de um nó. Em vez disso, seu valor pode ser acessado diretamente, por referência a uma variável.

O campo do próximo endereço do último nó na lista contém um valor especial, conhecido como NULL, que não é um endereço válido. Esse ponteiro nulo é usado para indicar o final de uma lista. Uma lista é chamada Lista Vazia ou Lista Nula caso não tenha nós ou tenha apenas um nó sentinela. O valor do ponteiro externo para esta lista é o ponteiro nulo. Uma lista pode ser inicializada com uma lista vazia.

[2] O endereço do primeiro nó pode ser encapsulado para facilitar possíveis futuras operações sobre a lista sem a necessidade de se conhecer sua estrutura interna. O primeiro elemento e o último nós são muitas vezes chamados de *Sentinela*.



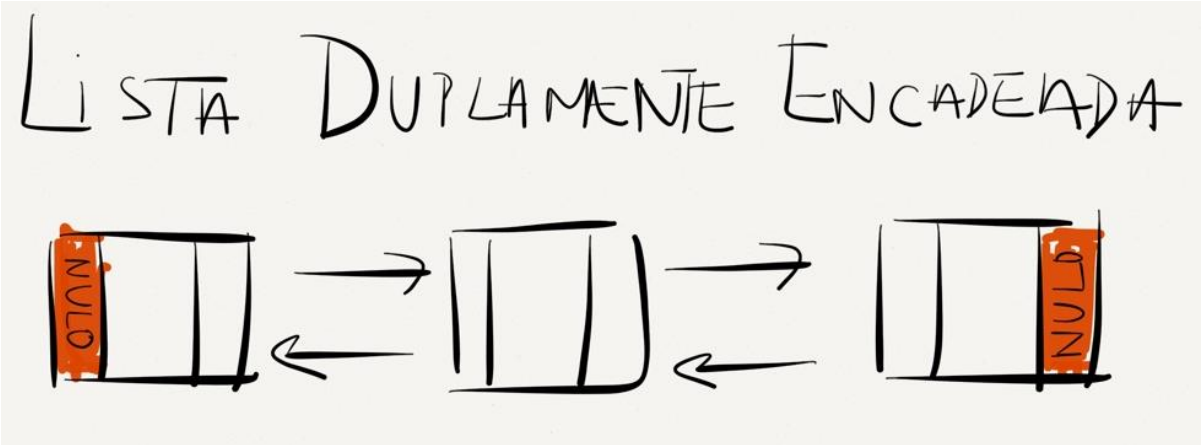
Suponha que seja feita uma mudança na estrutura de uma lista linear, de modo que o campo *próximo* no último nó contenha um ponteiro de volta para o primeiro nó, em vez de um ponteiro nulo. **Esse tipo de lista é chamado Lista Circular (ou Fechada[3]), i.e., a partir de qualquer ponto, é possível atingir qualquer outro ponto da lista.** ***Certinho, até agora?***



Observe que uma Lista Circular não tem um primeiro ou último nó natural. **Precisamos, portanto, estabelecer um primeiro e um último nó por convenção.** Uma convenção útil é permitir que o

[3] Se Listas Circulares são conhecidas como Listas Fechadas, as Listas Abertas são todas aquelas que são Não-Circulares. Por fim: da mesma forma que há Listas Circulares Simples, há também Listas Circulares Duplas. Nesse caso, o ponteiro anterior do primeiro elemento aponta para o último elemento e o ponteiro posterior do último elemento aponta para o primeiro elemento.

ponteiro externo para a lista circular aponte para o último nó, e que o nó seguinte se torne o primeiro nó. Assim podemos incluir ou remover um elemento convenientemente a partir do início ou do final de uma lista.



Embora uma lista circularmente ligada tenha vantagens sobre uma lista linear, ela ainda apresenta várias deficiências. Não se pode atravessar uma lista desse tipo no sentido contrário nem um nó pode ser eliminado de uma lista circularmente ligada sem se ter um ponteiro para o nó antecessor. **Nos casos em que tais recursos são necessários, a estrutura de dados adequada é uma lista duplamente ligada**.

Cada nó numa lista desse tipo contém dois ponteiros, um para seu predecessor e outro para seu sucessor. Na realidade, no contexto de listas duplamente ligadas, os termos *predecessor* e *sucessor* não fazem sentido porque a lista é totalmente simétrica. As listas duplamente ligadas podem ser lineares ou circulares e podem conter ou não um nó de cabeçalho.

Podemos considerar os nós numa lista duplamente ligada como consistindo em três campos: um campo *info* que contém as informações armazenadas no nó, e os campos *left* e *right*, que contêm ponteiros para os nós em ambos os lados. **Dado um ponteiro para um elemento, pode-se acessar os elementos adjacentes e, dado um ponteiro para o último elemento, pode-se percorrer a lista em ordem inversa**.

Existem cinco operações básicas sobre uma lista encadeada: Criação, em que se cria a lista na memória; Busca, em que se pesquisa nós na lista; Inclusão, em que se insere novos nós na lista em

uma determinada posição; Remoção, em que se elimina um elemento da lista; e, por fim, Destruição, em que se destrói a lista junto com todos os seus nós.

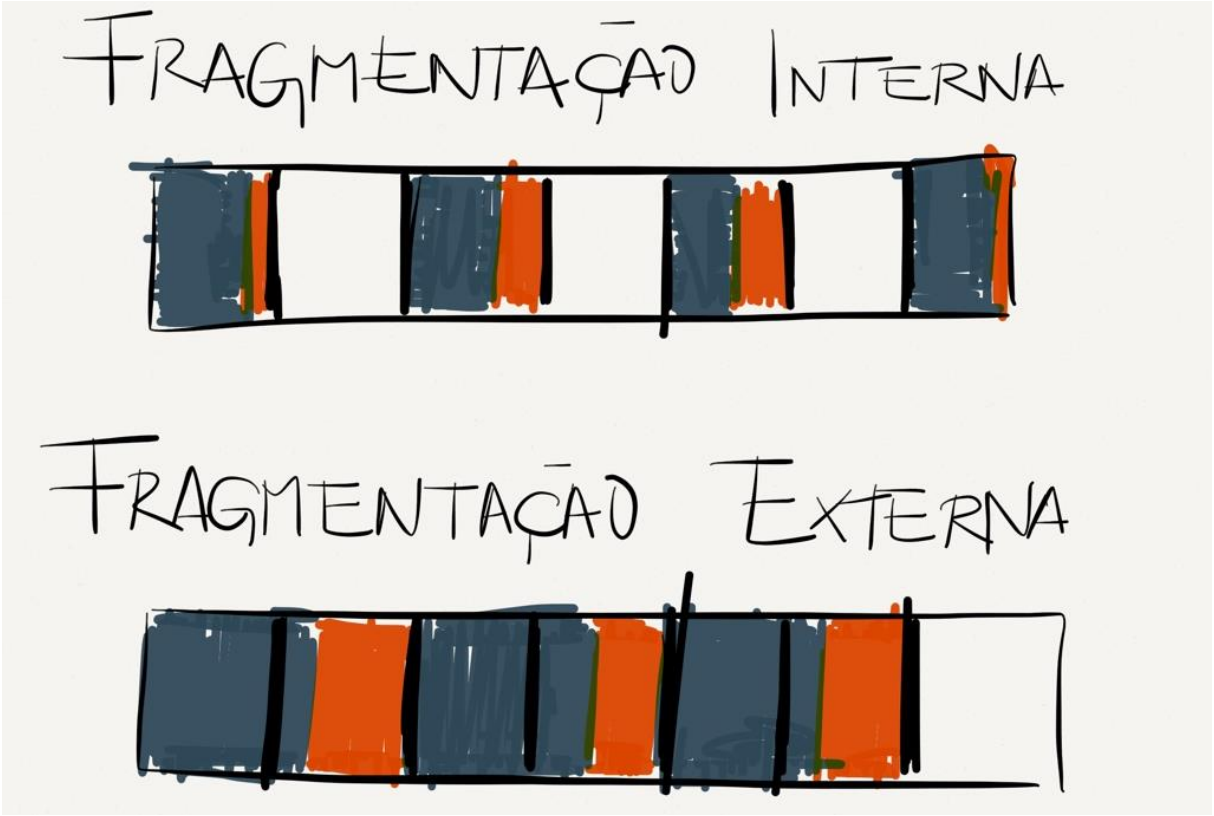
IMPORTANTE

Pilhas e Filas são subespécies de Listas. No entanto, cuidado na hora de responder questões! De maneira genérica, Pilhas e Filas podem ser implementadas como Listas. No entanto, elas possuem características particulares de uma lista genérica. Ok?

Precisamos falar um pouco sobre Fragmentação! *O que é isso, professor?* **Galera, falou em fragmentação, lembrem-se de desperdício de espaço disponível de memória.** O fenômeno no qual existem vários blocos disponíveis pequenos e não-contíguos é chamado fragmentação externa porque o espaço disponível é desperdiçado fora dos blocos alocados.

Esse fenômeno é o oposto da fragmentação interna, no qual o espaço disponível é desperdiçado dentro dos blocos alocados, como apresenta a imagem abaixo. Sistemas Operacionais possuem uma estrutura de dados que armazena informações sobre áreas ou blocos livres (geralmente uma lista ou tabela). **Uma lista encadeada elimina o problema da fragmentação externa. *Por que?***

Porque mantém os arquivos, cada um, como uma lista encadeada de blocos de disco. Dessa forma, uma parte de cada bloco é usada como ponteiro para o próximo bloco e o restante do bloco é usado para dados. Uma vantagem desse tipo de alocação é que o tamanho do arquivo não precisa ser conhecido antes de sua criação, já que cada bloco terá um ponteiro para o próximo bloco.



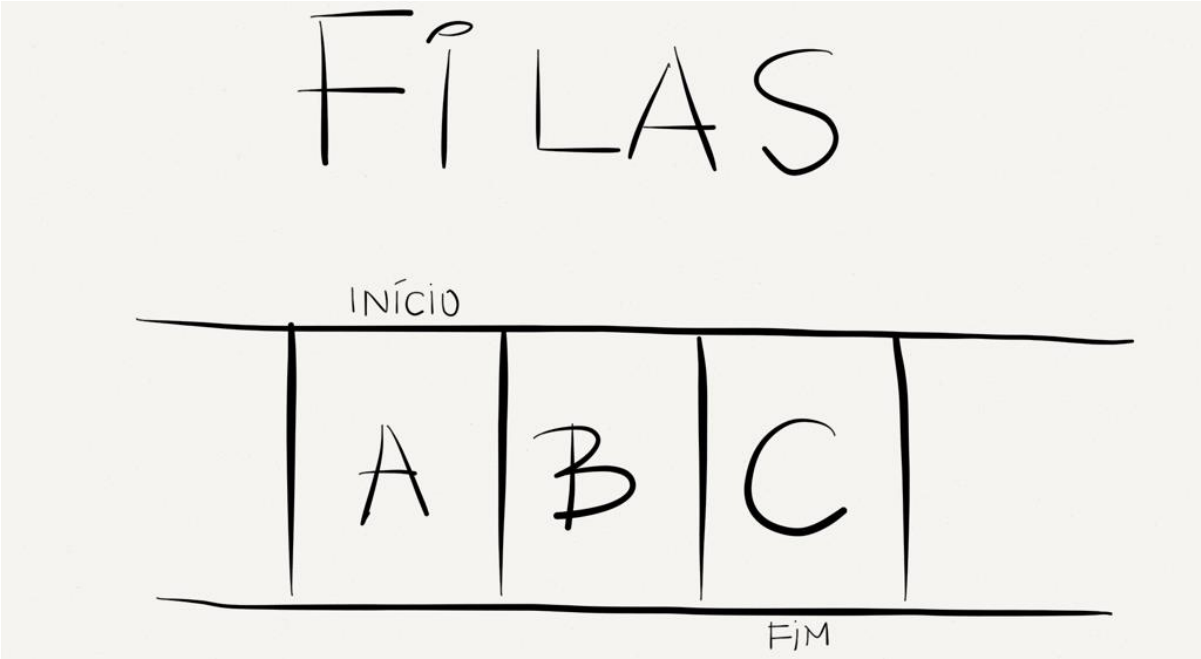
Galera... e o acesso a uma lista? **A Lista é uma estrutura de acesso sequencial, i.e., é preciso percorrer nó por nó para acessar um dado específico**. Logo, é proporcional ao número de elementos – Acesso O(n). *E os Vetores?* Eles são uma estrutura de acesso direto, i.e., pode-se acessar um elemento diretamente. Portanto, não precisa percorrer elemento por elemento (Acesso O(1))[4].

[4] No Acesso Sequencial: quanto mais ao fim, maior o tempo para acessar; no Acesso Direto: todos os elementos são acessados no mesmo tempo.

4 - Estruturas de dados: Filas

INCIDÊNCIA EM PROVA: média

Uma fila é um conjunto ordenado de itens a partir do qual podem-se eliminar itens numa extremidade (chamada início da fila) e no qual podem-se inserir itens na outra extremidade (chamada final da fila). Também conhecida como Lista FIFO (First In First Out), basta lembrar de uma fila de pessoas esperando para serem atendidas em um banco, i.e., o primeiro a entrar é o primeiro a sair.



Quando um elemento é colocado na fila, ele ocupa seu lugar no fim da fila, como um aluno recém-chegado que ocupa o final da fileira. O elemento retirado da fila é sempre aquele que está no início da fila, como o aluno que se encontra no começo da fileira e que esperou mais tempo. **As operações básicas são Enqueue (Enfileirar) e Dequeue (Desenfileirar). As Filas possuem início (ou cabeça) e fim (ou cauda).**

É bom salientar outro conceito importante: Deque (Double Ended Queue)! **É também conhecida como Filas Duplamente Encadeadas e permite a eliminação e inserção de itens em ambas as extremidades**. Ademais, elas permitem algum tipo de priorização, visto que é possível inserir elementos de ambos os lados. Assim sendo, é comum em sistemas distribuídos!

Sistemas distribuídos sempre necessitam que algum tipo de processamento seja mais rápido, por ser mais prioritário naquele momento, deixando outros tipos mais lentos ou em fila de espera, por não requerem tanta pressa. **Ele pode ser entendido como uma extensão da estrutura de dados Fila.** Agora uma pergunta: *Qual a diferença entre uma lista duplamente encadeada e um deque?*

Pessoal, um deque gerencia elementos como um vetor, fornece acesso aleatório e tem quase a mesma interface que um vetor. Ele se diferencia de uma lista duplamente encadeada, entre outras coisas, por essa não fornecer acesso aleatório aos elementos, i.e., para acessar o quinto elemento, você deve navegar pelos quatro primeiros elementos – logo a lista é mais lenta nesse sentido. *Bacana?*

(SEED-PR - 2021) Na estrutura de dados denominada FILA,
a) o último elemento a ser inserido será o primeiro a ser retirado.
b) o primeiro elemento a ser inserido será o primeiro a ser retirado: adiciona-se item no fim e remove-se item do início.
c) os elementos de um mesmo tipo de dado estão organizados de maneira sequencial e ordenada.
d) os elementos não estão necessariamente armazenados sequencialmente na memória por ordem descrente de valores.
e) os elementos são formados de índices em duas dimensões: linhas e colunas.

Comentários: Pessoal, conforme vimos em uma FILA o primeiro elemento a ser inserido será o primeiro a ser retirado (First in First out).

(Letra B).

(IF-RR – 2021) Existe uma estrutura de dados que representa um conjunto ordenado de elementos e cujas operações se baseiam no princípio FIFO (First-In, First-Out), ou seja, o primeiro elemento que entra é o primeiro a sair. A respectiva estrutura de dados é a Fila.

Comentários: Perfeito! Uma fila é um conjunto ordenado de itens a partir do qual podem-se eliminar itens numa extremidade (chamada início da fila) e no qual podem-se inserir itens na outra extremidade (chamada final da fila).

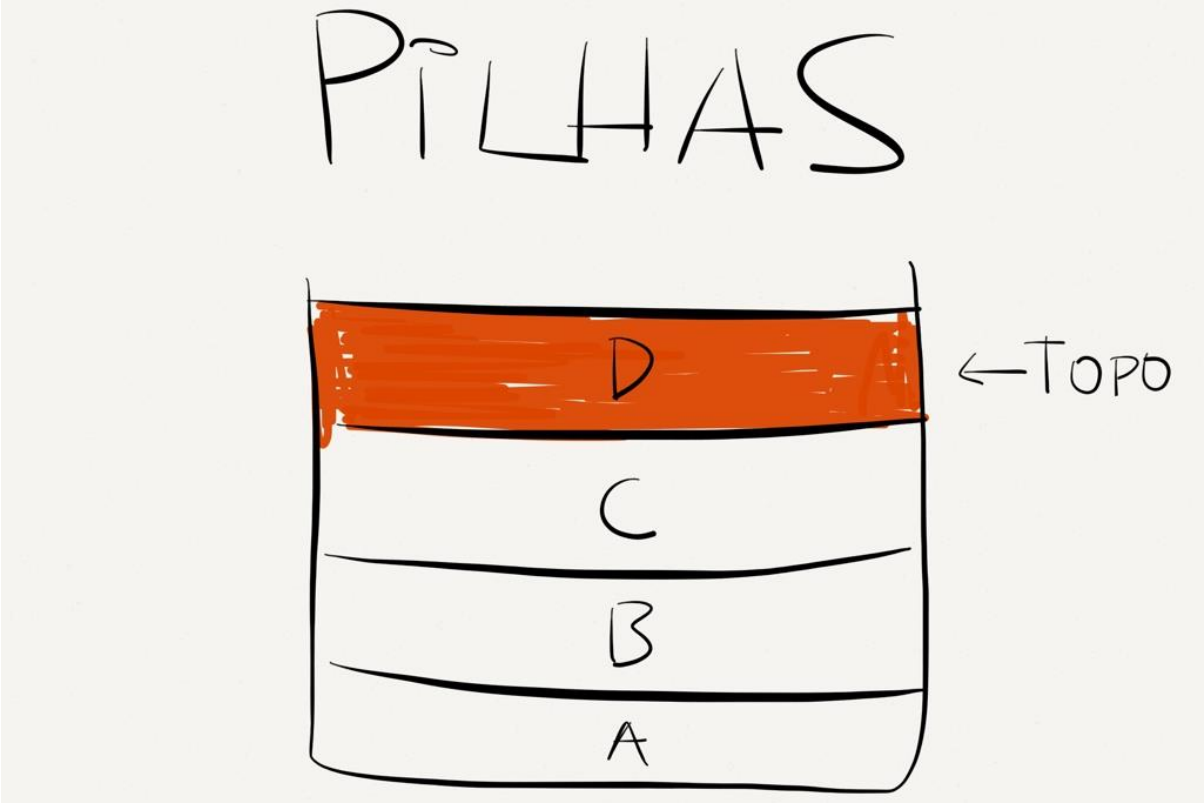
(CORRETO).

4 - Estruturas de dados: Pilhas

INCIDÊNCIA EM PROVA: média

A Pilha é um conjunto ordenado de itens no qual novos itens podem ser inseridos e eliminados em uma extremidade chamada *topo*. Novos itens podem ser colocados no topo da pilha (tornando-se o novo primeiro elemento) ou os itens que estiverem no topo da pilha poderão ser removidos (tornando-se o elemento mais abaixo o novo primeiro elemento).

Também conhecida como Lista LIFO (Last In First Out), basta lembrar de uma pilha de pratos esperando para serem lavados, i.e., o último a entrar é o primeiro a sair. A ordem em que os pratos são retirados da pilha é o oposto da ordem em que eles são colocados sobre a pilha e, como consequência, apenas o prato do topo da pilha está acessível.



As Pilhas oferecem três operações básicas: *push*, que insere um novo elemento no topo da pilha; *pop*, que remove um elemento do topo da pilha; e *top* (também conhecida como *check*), que acessa e consulta o elemento do topo da pilha. Pilhas podem ser implementadas

por meio de Vetores (Pilha Sequencial - Alocação Estática de Memória) ou Listas (Pilha Encadeada - Alocação Dinâmica de Memória).

(SEED-PR-2021) Em determinada estrutura de dados, os valores seguem a regra segundo a qual o último a entrar é o primeiro a sair.

a) PILHA
b) FILA
c) LISTA ENCADEADA
d) LISTA DUPLAMENTE ENCADEADA
e) MATRIZ

Comentários: As pilhas são conhecidas como Lista LIFO (Last In First Out), ou seja, o último a entrar é o primeiro a sair.

(LETRA A).

(Prefeitura de Jaru - 2019) As operações do tipo LIFO são típicas da estrutura de dados denominada Matriz.

Comentários: Pessoal, conforme vimos as pilhas tipicamente implementam as ordenações LIFO (Last In First Out).

(ERRADO).

(Câmara Municipal de Taboão da Serra - 2019) Considerando uma estrutura de dados do tipo "lista", se tanto as operações de inserção quanto as operações de remoção são realizadas somente em um de seus extremos, então pode-se afirmar que essa estrutura recebe o nome de pilha.

Comentários: Pessoal, conforme vimos se consideramos o tipo de estrutura de lista, com operações de inserção e operações de remoção sendo realizadas somente em um de seus extremos, então podemos afirmar que essa estrutura recebe o nome de pilha

(CORRETO).

(CRM-MT - 2021) Assinale a alternativa que indica corretamente o nome da estrutura de dados que é unidimensional e que possui a característica de permitir acesso imediato apenas ao elemento mais recentemente inserido nela.

A) Matriz
B) Fila
C) Pilha
D) Árvore binária

Comentários: Pessoal, falou em ordenação LIFO (Last In First Out), ou seja, permitir acesso imediato apenas ao elemento mais recentemente inserido, estamos falando da estrutura de dados PILHA.

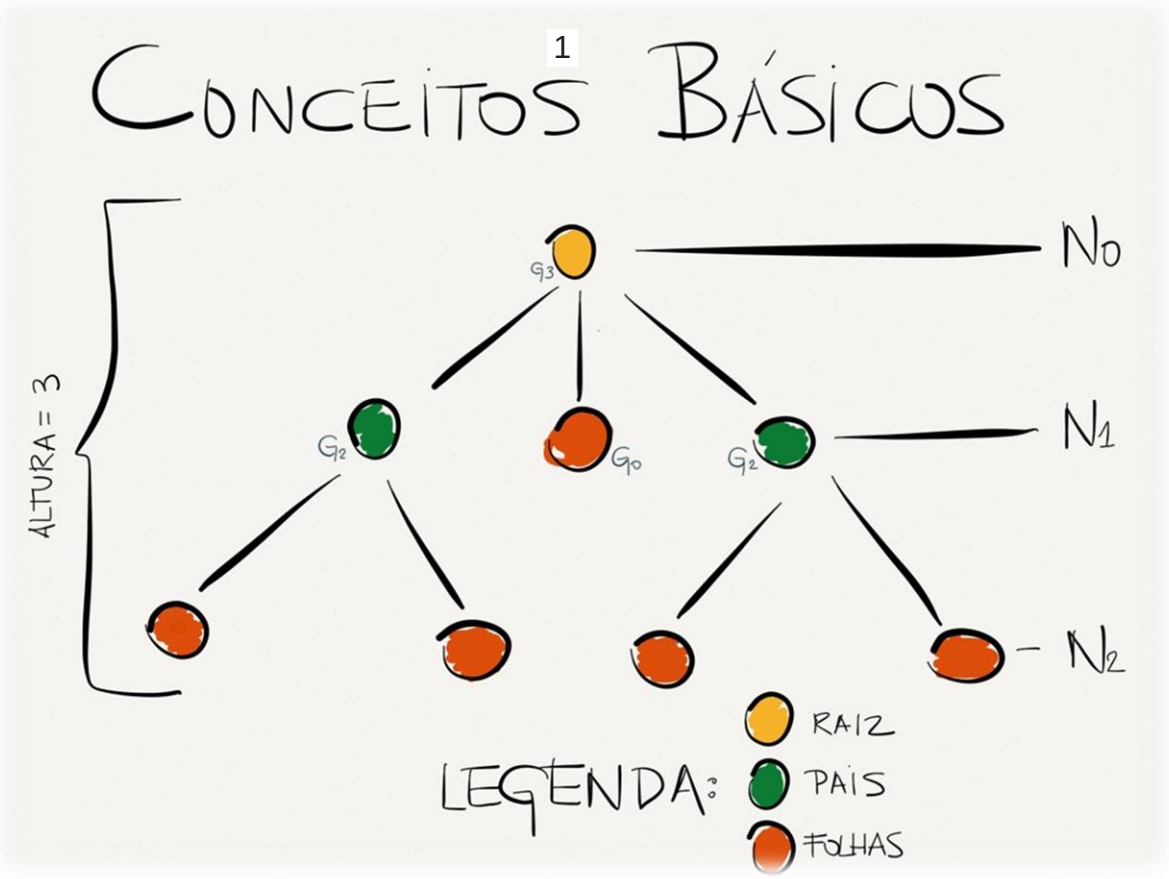
(Letra C).

1

5 - Estruturas de dados: Árvores

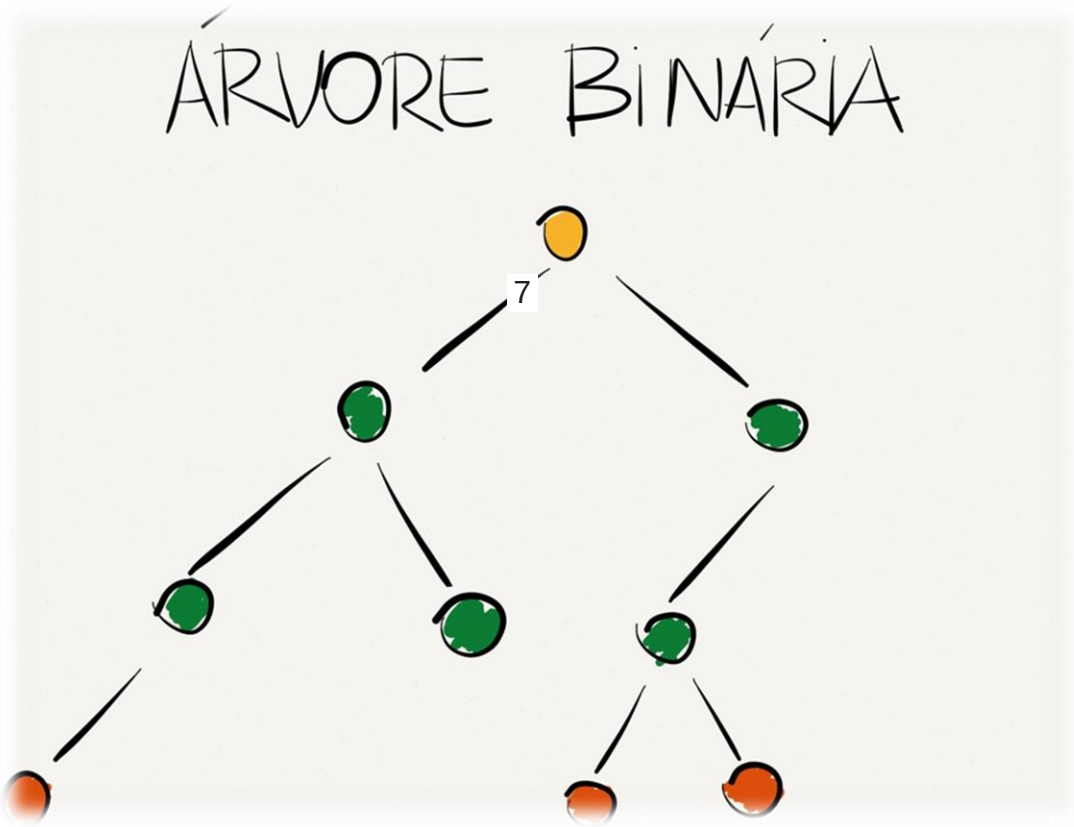
INCIDÊNCIA EM PROVA: média

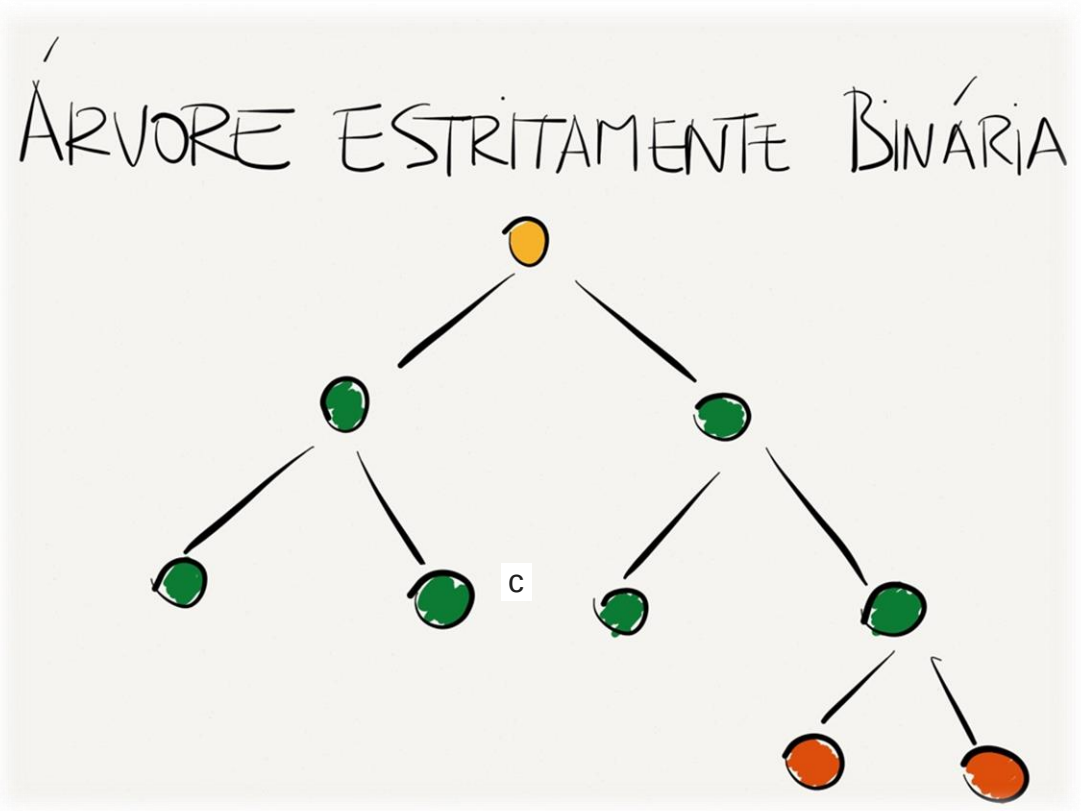
Uma árvore é uma estrutura de dados hierárquica (não-linear) composta por um conjunto finito de elementos com um único elemento raiz, com zero ou mais sub-árvores ligadas a esse elemento raiz. Como mostra a imagem abaixo, há uma única raiz, em amarelo. Há também nós folhas, em vermelho e seus pais, em verde. Observem ainda os conceitos de Altura, Grau e Nível de uma árvore.

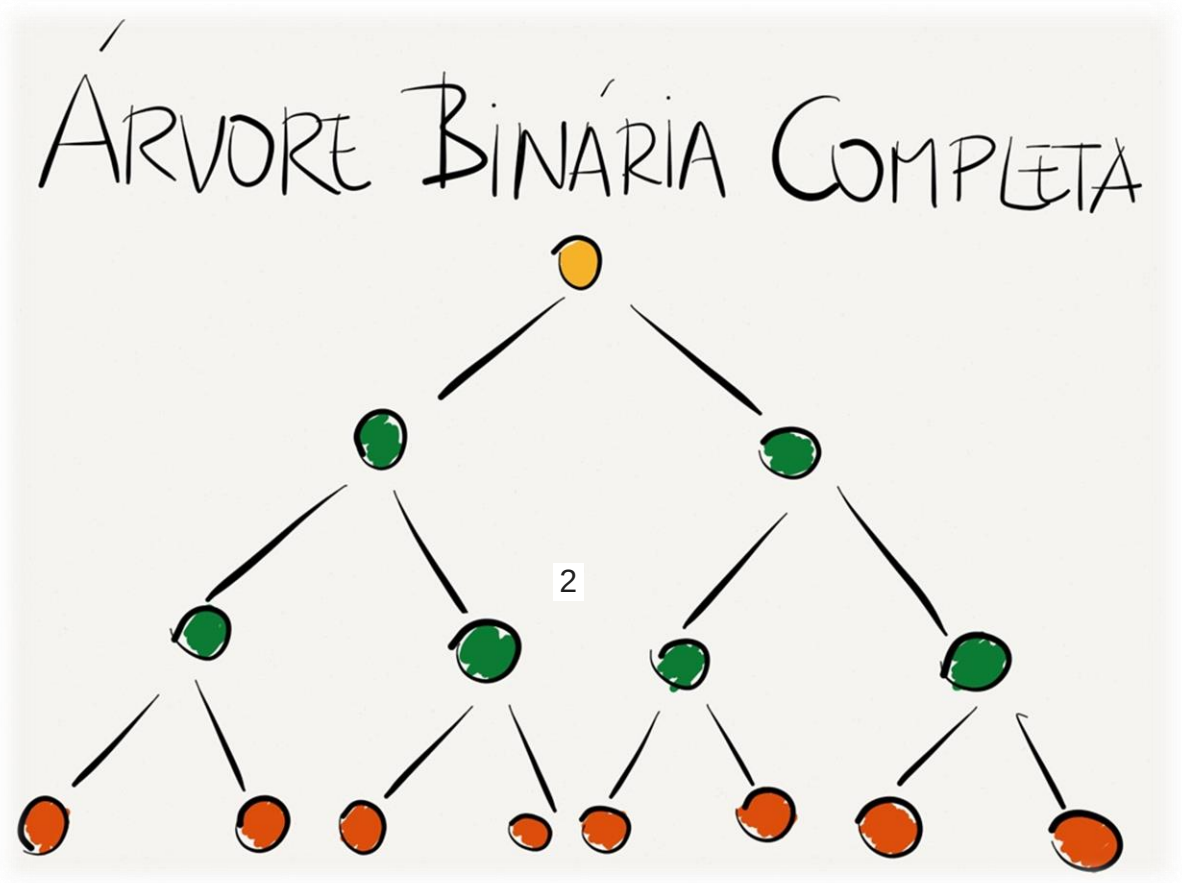


O Grau informa a quantidade de filhos de um determinado nó! A Raiz tem Nível 0 (excepcionalmente, alguns autores consideram que tem Nível 1) e o nível de qualquer outro nó na árvore é um nível a mais que o nível de seu pai. Por fim, a Altura é a distância entre a raiz e seu descendente mais afastado. **Dessas informações, podemos concluir que toda folha tem Grau 0.**

Existe um tipo particular de árvore chamado: Árvore Binária! *O que é isso?* É uma estrutura de dados hierárquica em que todos os nós têm grau 0, 1 ou 2. **Já uma Árvore Estritamente Binária é aquela em que todos os nós têm grau 0 ou 2.** E uma Árvore Binária Completa é aquela em que todas as folhas estão no mesmo nível, como mostram as imagens abaixo.




ÁRVORE ESTRITAMENTE BINÁRIA
C



Uma Árvore Binária Completa com *x* folhas conterá sempre (2x – 1) nós. Observem a imagem acima e façam as contas: 2*8 – 1 = 15 nós! **Uma árvore binária completa de altura *h* e nível *n* contém (2^h-1) ou (2^{n+1}-1) nós e usa-se (2^n), para calcular a quantidade de nós em determinado nível.** Na imagem acima, há uma árvore de h = 4 e n = 3; logo, existem 2^{3+1} -1 = 15 nós no total; e no Nível 3, existe 2^3 = 8 nós.

Vamos falar um pouco agora sobre Árvore de Busca Binária! Trata-se de uma Uma estrutura de dados vinculada, baseada em nós, onde cada nó contém uma chave e duas subárvores à esquerda e a direita. Para todos nós, a chave da subárvore esquerda deve ser menor que a chave desse nó, e a chave da subárvore direita deve ser maior. *Beleza?*

(Prefeitura de Itapemirim - 2019) Estruturas de dados são métodos para armazenagem de dados de forma eficiente. Das estruturas abaixo, a utilizada pelos bancos de dados hierárquicos é:

a) PILHA
b) FILA
c) QUADRADO
d) TABELA DE DISPERSÃO
e) ÁRVORE

Comentários: Questão muito interessante, os bancos de dados hierárquicos utilizam a estrutura de dados: árvore. O banco de dados hierárquico conecta os registros de forma estruturada através de ligações onde cada registro tenha só um possuidor, constituindo uma estrutura de dados em árvore. b

(LETRA E).

(UFCSPA – 2018) Uma árvore estritamente binária é caracterizada por ter:

a) cada nó associado a um valor que pode ser falso ou verdadeiro
b) todos os seus níveis com a mesma quantidade de nós
c) no máximo dois nós-filhos por nó-pai
d) uma estrutura balanceada
e) uma profundidade equivalente a uma potência de 2

Comentários: Uma Árvore Estritamente Binária é aquela em que todos os nós têm grau 0 ou 2.

(LETRA C).

Todas estas subárvores devem qualificar-se como árvores binárias de busca. O pior caso de tempo de complexidade para a pesquisa em uma árvore binária de busca é a altura da árvore, isso pode ser tão pequeno como O(log n) para uma árvore com n elementos. **Galera, abaixo nós vamos ver como árvores podem ser representadas e o conceito de árvore binária de busca ficará mais clara!**

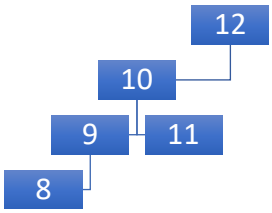
Como representamos árvores? **Podemos representar uma árvore como um conjunto de parênteses aninhados.** Nessa notação, (P (F_1)(F_2)) significa que P, F_1, F_2 são nós e que F_1, F_2, são filhos do pai P. Ao transcrever isso para o desenho hierárquico de uma árvore, lemos da esquerda

para a direita. Agora suponhamos que F_1 tem dois filhos N_1 e N_2. Logo, reescrevemos a subárvore de F_1 como $(F_1\ (N_1)(N_2))$.

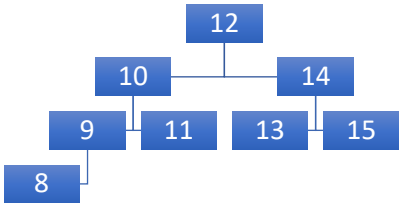
Podemos, então, substituir F_1 por $(F_1(N_1)(N_2))$. Ao final, ficará $(P(F_1(N_1)(N_2))(F_2))$. E assim por diante. **Temos então que o primeiro elemento é a raiz e sempre que tivermos um parêntese, teremos uma nova subárvore**. Vamos exemplificar: **(12(10(9(8))(11))(14(13)(15)))**. *Como ficaria, professor?* Sabemos que 12 será a raiz dessa árvore e, a partir daí, criamos a árvore da esquerda para direita.

Observem o parêntese após o 12! Notem que ele só é fechado após o 11: **(12(10(9(8))(11))**. Isso significa que tudo que está em vermelho é subárvore da esquerda da raiz 12 – e o restante **(14(13)(15))** é subárvore da direita da raiz 12. Observem o parêntese após o 10! Notem que ele só é fechado após o 8: **10(9(8))**. **Isso significa que tudo que está em amarelo é subárvore da esquerda da raiz 10.**

E o restante **(11)** é subárvore da direita da raiz 10. Observem o parêntese após o 9! Notem que ele só é fechado após o 8: **9(8)**. **Isso significa que tudo que está em verde é subárvore da esquerda da raiz 9 – e o restante...** *que restante, professor?* Pois é, não tem restante! Logo, 9 não tem subárvore da direita. Vejam abaixo como ficou e vamos analisar o outro lado.

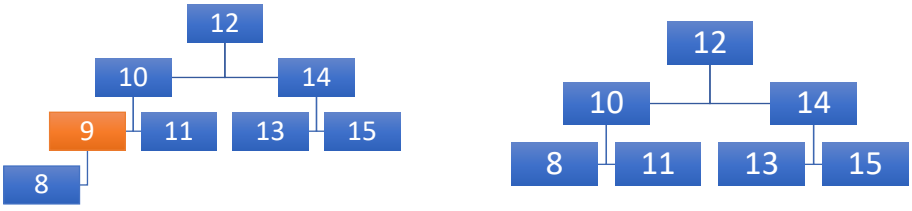


A subárvore da direita da raiz 12 tem raiz 14 e tem dois filhos: na esquerda, 13 e na direita 15. Fim! Galera, eu sei que parece complicado, mas leiam e pratiquem umas três vezes – de preferência no papel - que vocês internalizam tranquilamente esse conteúdo. É chatinho, mas não é difícil! Vejam abaixo como ficou o resultado final e vamos seguir em frente...



Bem, pessoal! **Dito isso, vamos analisar agora como ficaria para remover um nó desta árvore.** Existem três possibilidades para realizar essa operação: (1) remover um nó que não tem filhos; (2) remover um nó que tem apenas um filho; (3) e remover um nó que tenha dois filhos. O primeiro caso é muito simples: basta retirar o nó desejado e ponto final. Vejamos como fica:

No segundo caso, basta retirar o nó da árvore e conectar seu único filho (e sua subárvore, se houver) diretamente ao pai do nó removido. Vejamos:



Já o último caso, nós podemos utilizar duas estratégias. Você pode escolher qual deseja utilizar em uma situação específica. Vejamos:

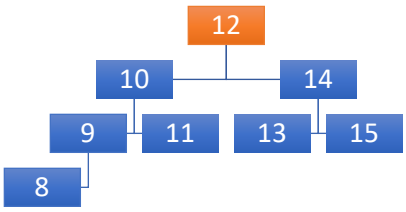
ESTRATÉGIA 1

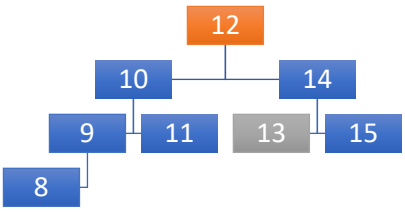
PASSO 1: IDENTIFIQUE O ELEMENTO QUE VOCÊ DESEJA RETIRAR DA ÁRVORE (EM VERMELHO)

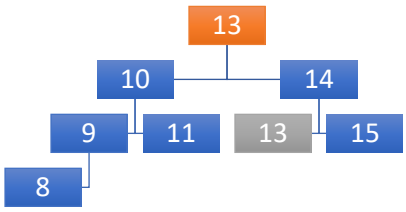
PASSO 2: IDENTIFIQUE O MENOR ELEMENTO DE TODA SUBÁRVORE À DIREITA DO NÓ IDENTIFICADO NO PASSO 1 (EM VERDE)

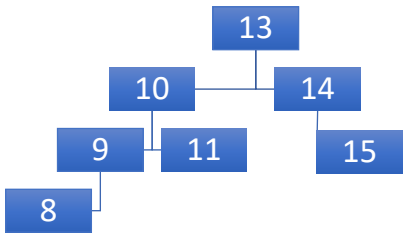
PASSO 3: COPIE O VALOR DO NÓ IDENTIFICADO NO PASSO 2 PARA O NÓ IDENTIFICADO NO PASSO 1

PASSO 4: REMOVA O ELEMENTO IDENTIFICADO NO PASSO 2.









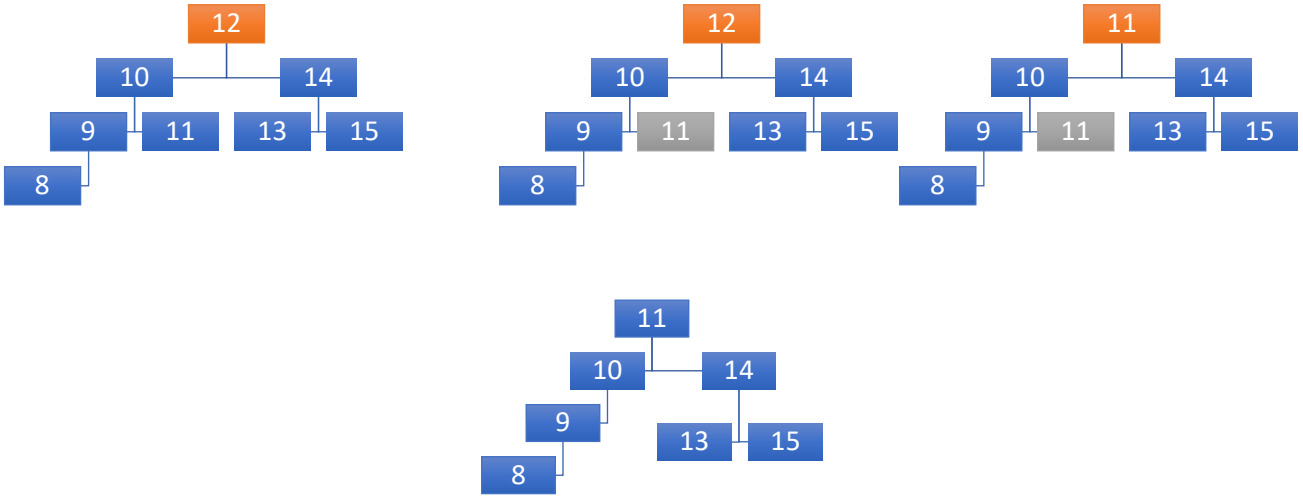
ESTRATÉGIA 2

PASSO 1: IDENTIFIQUE O ELEMENTO QUE VOCÊ DESEJA RETIRAR DA ÁRVORE (EM VERMELHO)

PASSO 2: IDENTIFIQUE O MAIOR ELEMENTO DE TODA SUBÁRVORE À ESQUERDA DO NÓ IDENTIFICADO NO PASSO 1 (EM VERDE)

PASSO 3: COPIE O VALOR DO NÓ IDENTIFICADO NO PASSO 2 PARA O NÓ IDENTIFICADO NO PASSO 1

PASSO 4: REMOVA O ELEMENTO IDENTIFICADO NO PASSO 2.



Por fim, como seria a representação por parênteses aninhados? Na primeira estratégia, temos: (13(10(9(8))(11))(14(15))); na segunda, temos (11(10(9(8)))(14(13)(15))).

Galera, em uma Árvore de Busca Binária, podemos fazer três percursos: **pré-ordem, in-ordem e pós-ordem (esses prefixos são em relação a raiz).** É interessante notar que, quando se faz um percurso em ordem em uma árvore binária de busca, os valores dos nós aparecem em ordem crescente. A operação "Percorre" tem como objetivo percorrer a árvore numa dada ordem, enumerando os seus nós.

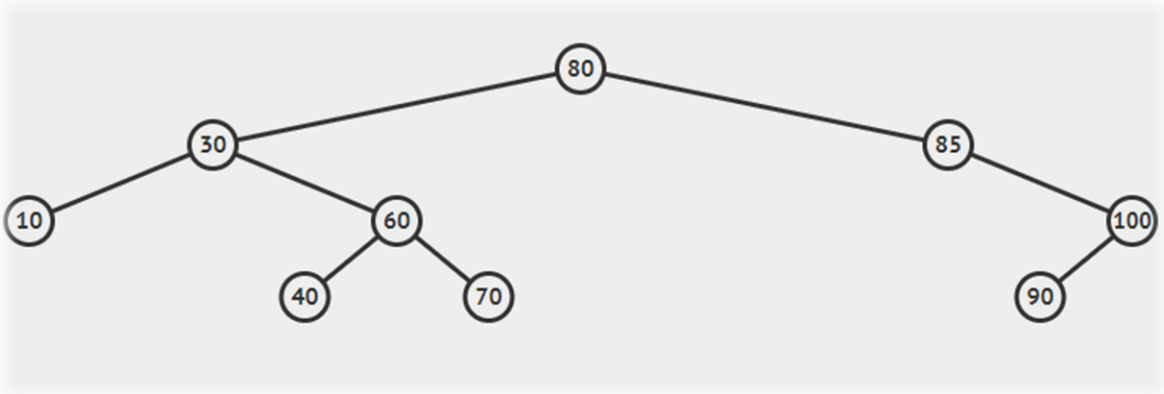
Quando um nó é enumerado, diz-se que ele foi "visitado". Vamos ver agora esses três percursos:

- **Pré-Ordem (ou Profundidade):** visita a raiz; percorre a subárvore esquerda em pré-ordem; percorre a subárvore direita em pré-ordem.

- **In-Ordem (ou Simétrica):** percorre a subárvore esquerda em in-ordem; visita a raiz; percorre a subárvore direita em in-ordem.

- **Pós-Ordem:** percorre a subárvore esquerda em pós-ordem; percorre a subárvore direita em pós-ordem; visita a raiz.

Vamos ver um exemplo:



Como ler essa árvore em Pré-Ordem? Vamos tentar...

//Percorrendo a Árvore em Pré-Ordem:

1. Visite a Raiz: {**80**};
2. Percorra a subárvore esquerda em pré-ordem:
    1. Visite a Raiz: {**30**};
    2. Percorra a subárvore esquerda em pré-ordem:
        1. Visite a Raiz: {**10**};
        2. Percorra a subárvore esquerda em pré-ordem: {Vazio}
        3. Percorra a subárvore direita em pré-ordem: {Vazio}
    3. Percorra a subárvore direita em pré-ordem:
        1. Visite a Raiz: {**60**};

2. Percorra a subárvore esquerda em pré-ordem:

    1. Visite a Raiz: {**40**}

    2. Percorra a subárvore esquerda em pré-ordem: {Vazio}

    3. Percorra a subárvore direita em pré-ordem: {Vazio}

2. Percorra a subárvore direita em pré-ordem:

    1. Visite a Raiz: {**70**}

    2. Percorra a subárvore esquerda em pré-ordem: {Vazio}

    3. Percorra a subárvore direita em pré-ordem: {Vazio}

3. Percorra a subárvore direita em pré-ordem:

    1. Visite a Raiz: {**85**};

    2. Percorra a subárvore esquerda em pré-ordem: {Vazio}

    3. Percorra a subárvore direita em pré-ordem: {Vazio}

        1. Visite a Raiz: {**100**};

        2. Percorra a subárvore esquerda em pré-ordem:

            1. Visite a Raiz: {**90**};

            2. Percorra a subárvore esquerda em pré-ordem: {Vazio}

            3. Percorra a subárvore direita em pré-ordem: {Vazio}

        3. Percorra a subárvore direita em pré-ordem: {Vazio}

RESULTADO: 80, 30, 10, 60, 40, 70, 85, 100, 90

```
//Percorrendo a Árvore em In-Ordem:
```

1. Percorra a subárvore esquerda em in-ordem:
    1. Percorra a subárvore esquerda em in-ordem:
        1. Percorra a subárvore esquerda em in-ordem: {Vazio}
        2. Visite a Raiz: {**10**}
        3. Percorra a subárvore direita em in-ordem: {Vazio}
    2. Visite a Raiz: {**30**}
    3. Percorra a subárvore direita em in-ordem:
        1. Percorra a subárvore esquerda em in-ordem:
            1. Percorra a subárvore esquerda em in-ordem: {Vazio}
            2. Visite a Raiz: {**40**}
            3. Percorra a subárvore direita em in-ordem: {Vazio}
        2. Visite a Raiz: {**60**}
        3. Percorra a subárvore direita em in-ordem:
            1. Percorra a subárvore esquerda em in-ordem: {Vazio}
            2. Visite a Raiz: {**70**}
            3. Percorra a subárvore direita em in-ordem: {Vazio}
2. Visite a Raiz: {**80**}
3. Percorra a subárvore direita em in-ordem:
    1. Percorra a subárvore esquerda em in-ordem: {Vazio}
    2. Visite a Raiz: {**85**}
    3. Percorra a subárvore direita em in-ordem:

1. Percorra a subárvore esquerda em in-ordem:
    1. Percorra a subárvore esquerda em in-ordem: {Vazio}
    2. Visite a Raiz: {**90**}
        3. Percorra a subárvore direita em in-ordem: {Vazio}
2. Visite a Raiz: {**100**}
    3. Percorra a subárvore direita em in-ordem: {Vazio}

RESULTADO: 10, 30, 40, 60, 70, 80, 85, 90, 100.

//Percorrendo a Árvore em Pós-Ordem:

1. Percorra a subárvore esquerda em pós-ordem:
    1. Percorra a subárvore esquerda em pós-ordem:
        1. Percorra a subárvore esquerda em pós-ordem: {Vazio}
        2. Percorra a subárvore direita em pós-ordem: {Vazio}
        3. Visite a Raiz: {**10**}
    2. Percorra a subárvore direita em pós-ordem:
        1. Percorra a subárvore esquerda em pós-ordem: {Vazio}
            1. Percorra a subárvore esquerda em pós-ordem: {Vazio}
            2. Percorra a subárvore direita em pós-ordem: {Vazio}
            3. Visite a Raiz: {**40**}
        2. Percorra a subárvore direita em pós-ordem:

1. Percorra a subárvore esquerda em pós-ordem: {Vazio}

2. Percorra a subárvore direita em pós-ordem: {Vazio}

3. Visite a Raiz: {**70**}

3. Visite a Raiz: {**60**}

3. Visite a Raiz: {**30**}

2. Percorra a subárvore direita em pós-ordem:

1. Percorra a subárvore esquerda em pós-ordem: {Vazio}

2. Percorra a subárvore direita em pós-ordem:

1. Percorra a subárvore esquerda em pós-ordem:

1. Percorra a subárvore esquerda em pós-ordem: {Vazio}

2. Percorra a subárvore direita em pós-ordem: {Vazio}

3. Visite a Raiz: {**90**}

2. Percorra a subárvore direita em pós-ordem: {Vazio}

3. Visite a Raiz: {**100**}

3. Visite a Raiz: {**85**}

3. Visite a Raiz: {**80**}

RESULTADO: 10, 40, 70, 60, 30, 90, 100, 85, 80.

Agora vamos falar um pouquinho sobre três tipos especiais e árvores: Árvore B, Árvore B+ e Árvore AVL. E aí, eu preciso bastante da atenção de vocês agora! Eu coloco esse assunto porque os editais pedem apenas Árvore e não especificam a profundidade do assunto. Com exceção da CESGRANRIO, é raro outras bancas cobrarem esse assunto na profundidade que veremos agora.

É um assunto mais difícil e que cai pouco em prova, portanto só recomendo seguir caso queiram realmente cercar todas as possibilidades. Galera, época de faculdade, segundo semestre, disciplina de Estrutura de Dados! O trabalho final da disciplina era construir um compactador! Isso mesmo! Uma espécie de WinZip, WinRar, etc. E a estrutura usada para compactar arquivos de índices era uma Árvore B.

Uma árvore B é uma maneira de armazenar grandes quantidades de dados de tal forma que você pode procurá-los e recuperá-los muito rapidamente. **As árvores B são a base da maioria dos bancos de dados modernos**. Como eles funcionam é um bocado complicado, mas eu vou contar uma historinha que talvez os ajude a entender melhor! Vamo lá...

Imagine que você está procurando um par de novos fones de ouvido. Você tem algumas abordagens. *Certo?* **Você poderia ir a todas as lojas do mundo até encontrar o produto que você procura**. Como você pode imaginar, esta seria uma maneira horrível de fazer compras. Em vez disso, você poderia ir em uma FNAC, porque você sabe que eles vendem eletrônicos.

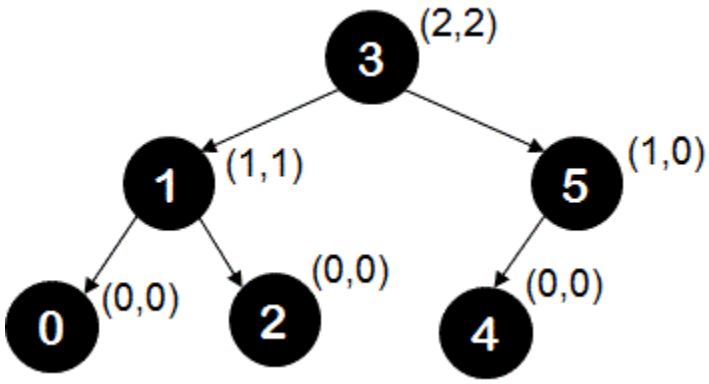
Uma vez que chegou à FNAC, você pode observar cada uma das descrições de corredor para ver onde os fones de ouvido são armazenados. Depois de encontrar o corredor correto, você pode escolher os fones de ouvido que você deseja. Ponto final! Observe como o objetivo desse processo é restringir o foco de uma pesquisa cada vez mais...

É assim que funcionam as Árvores B! Ao organizar os dados de forma específica, podemos aproveitá-las para que não desperdicemos nosso tempo buscando dados que tenham chance zero de armazenar os dados que estamos procurando. **E a árvore já é construída de maneira a organizar os dados da melhor forma possível.** *Bacana, pessoal?*

E qual a diferença de uma Árvore B para uma Árvore B+? Galera, a principal diferença é que, em uma Árvore B, as chaves e os dados podem ser armazenados tanto nos nós internos da árvore quanto nas folhas da árvore, **enquanto que em uma Árvore B+ as chaves podem ser armazenadas em qualquer nó, mas os dados só podem ser armazenados nas folhas**.

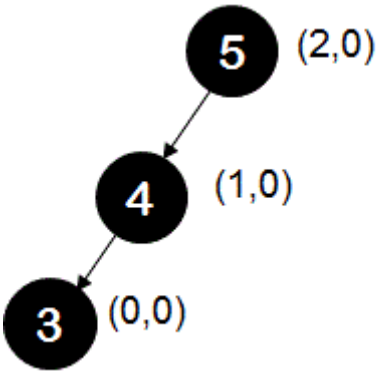
Por fim, vamos falar um pouco sobre Árvores AVL! Uma Árvore AVL (Adelson-Vesky e Landis) é uma Árvore Binária de Busca em que, para qualquer nó, a altura das subárvores da esquerda e da

direita não podem ter uma diferença maior do que 1, **portanto uma Árvore AVL é uma Árvore Binária de Busca autobalanceável. Calma que nós vamos ver isso em mais detalhes...**

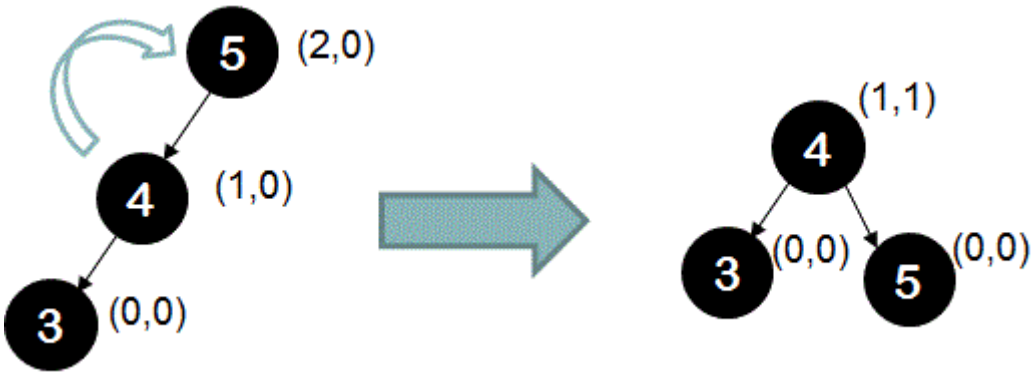


Observem o Nó 3: a altura da subárvore da esquerda é 2 e da subárvore da direita também é 2. Vejamos agora o Nó 1: a altura da subárvore da esquerda é 1 e da subárvore da direita também é 1. E o Nó 5: a altura da subárvore da esquerda é 1 e da subárvore da direita é 0 (visto que ela não existe). **Vocês podem ver todos os nós e não encontrarão subárvore da esquerda e direita com diferença maior que 1.**

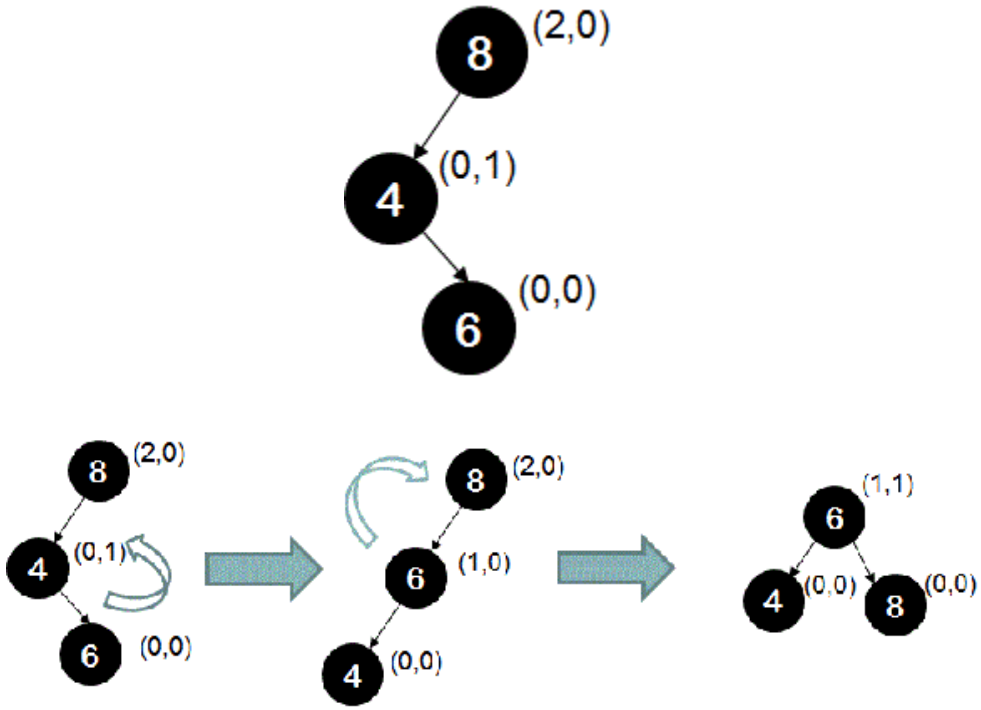
Uma Árvore AVL mantém seu equilíbrio de altura executando operações de rotação se algum dos nós violar a propriedade da diferença maior que 1. Exemplo 1: para a seguinte árvore, a propriedade da árvore AVL é violada no Nó 5, porque a subárvore esquerda possui altura 2, mas a subárvore da direita tem altura 0, então eles diferem em 2. *Entenderam?*



Essa diferença de 2 é construída nos dois ramos esquerdos - Ramo 5-4 e Ramo 4-3. *Concordam?* Se a diferença de 2 for construída por dois ramos esquerdos, chamamos esse caso de um Caso Esquerdo-Esquerdo (*Genial, né?)*. **Neste caso, realizamos uma rotação à direita no Ramo 5-4 como mostrado na imagem abaixo de forma a rebalancear a árvore.**



Já na imagem abaixo, a Árvore AVL tem sua propriedade violada no Nó 8. A altura da subárvore esquerda é maior do que a altura da subárvore direita em 2. Essa diferença de 2 é construída por um ramo esquerdo (Ramo 8-4) e um ramo direito (Ramo 4-6), logo é um Caso Esquerdo-Direito. Para restaurar o equilíbrio de altura, executamos uma rotação esquerda seguida de uma rotação direita. Vejam...



Então, galera, caso a árvore não esteja balanceada, é necessário seu balanceamento através de rotações. No Caso Esquerda-Esquerda, basta fazer uma rotação simples para direita no nó desbalanceado. No caso Esquerda-Direita, temos que fazer uma rotação dupla, i.e., **faz-se uma rotação para esquerda no nó filho e uma rotação para direita no nó desbalanceado.**

No caso Direita-Direita, basta fazer uma rotação simples para a esquerda no nó desbalanceado. No caso Direita-Esquerda, temos que fazer uma rotação dupla, i.e., faz-se uma rotação para direita no nó filho, seguida de uma rotação para esquerda no nó desbalanceado. *Bacana?* **Vocês entenderão isso melhor nos exercícios. Vamos ver agora a complexidade logarítmica dessas estruturas.**

ÁRVORE BINÁRIA DE BUSCA		
ALGORITMO	CASO MÉDIO	PIOR CASO
Espaço	O(n)	O(n)
Busca	O(log n)	O(n)
Inserção	O(log n)	O(n)
Remoção	O(log n)	O(n)

ÁRVORE B / ÁRVORE AVL		
ALGORITMO	CASO MÉDIO	PIOR CASO
Espaço	O(n)	O(n)
Busca	O(log n)	O(log n)
Inserção	O(log n)	O(log n)
Remoção	O(log n)	O(log n)

Quem quiser brincar de Árvore Binária de Busca

https://www.cs.usfca.edu/~galles/visualization/BST.html

Quem quiser brincar de Árvore AVL

https://www.cs.usfca.edu/~galles/visualization/AVLtree.html

(Prefeitura de Cunha Porã - 2021) Sobre o tema, Estrutura de Dados, analise as assertivas e assinale a alternativa correta.

I. Pilhas - São estruturas de dados do tipo LIFO (last-in first-out), onde o último elemento a ser inserido, será o primeiro a ser retirado. Assim, uma pilha permite acesso a apenas um item de dados - o último inserido. Para processar o penúltimo item inserido, deve-se remover o último.

II. Filas - São estruturas de dados do tipo FIFO (first-in first-out), onde o primeiro elemento a ser inserido, será o primeiro a ser retirado, ou seja, adiciona-se itens no fim e remove-se do início.

III. Lista linear - É uma estrutura de dados na qual elementos de um mesmo tipo de dado estão organizados de maneira sequencial. Não necessariamente, estes elementos estão fisicamente em sequência, mas a ideia é que exista uma ordem lógica entre eles.

IV. Árvore - É uma estrutura de dados que herda as características das topologias em árvore. Conceitualmente diferente das listas encadeadas, em que os dados se encontram numa sequência, nas árvores os dados estão dispostos de forma hierárquica. Uma árvore é formada por um conjunto de elementos que armazenam informações chamados nodos. Toda a árvore possui o elemento chamado raiz, que possui ligações para outros elementos denominados ramos ou filhos. Estes ramos podem estar ligados a outros elementos que também podem possuir outros ramos. O elemento que não possui ramos é conhecido como nó folha, nó terminal ou nó externo.

a) Apenas I e III estão corretas.
b) Apenas II e III estão corretas.
c) Apenas III e IV estão corretas.
d) Todas estão corretas.

Comentários: Questão com todas as alternativas corretas. Vejamos as descrições que estudamos:

Pilha - A Pilha é um conjunto ordenado de itens no qual novos itens podem ser inseridos e eliminados em uma extremidade chamada topo. (LIFO)

Fila – **Uma fila é um conjunto ordenado de itens a partir do qual podem-se eliminar itens numa extremidade (chamada início da fila) e no qual podem-se inserir itens na outra extremidade (chamada final da fila).** Também conhecida como Lista FIFO (First In First Out), basta lembrar de uma fila de pessoas esperando para serem atendidas em um banco, i.e., o primeiro a entrar é o primeiro a sair

Lista linear – Também conhecida como *Lista Encadeada Linear*, *Lista Ligada Linear* ou *Linked List*, trata-se de uma estrutura de dados dinâmica formada por uma sequência encadeada de elementos chamados nós, que contêm dois campos: **campo de informação** e **campo de endereço**. O primeiro armazena o real elemento da lista e o segundo contém o endereço do próximo nó da lista.

Árvore - **Uma árvore é uma estrutura de dados hierárquica (não-linear) composta por um conjunto finito de elementos com um único elemento raiz, com zero ou mais sub-árvores ligadas a esse elemento raiz.**

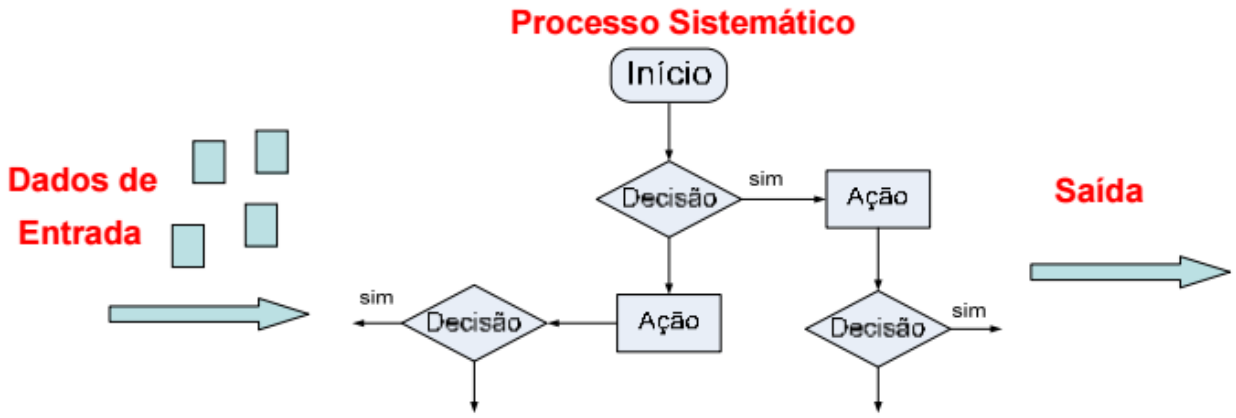
(LETRA D).

RESUMO DE ESTRUTURA DE DADOS

1 - Conceitos gerais

Começaremos definindo afinal de contas o que é uma estrutura de dados e por que estudamos estrutura de dados. Para você compreender bem o conceito de estrutura de dados precisamos entender o conceito de algoritmo como um processo sistemático. Ok?

Algoritmo como um Processo Sistemático



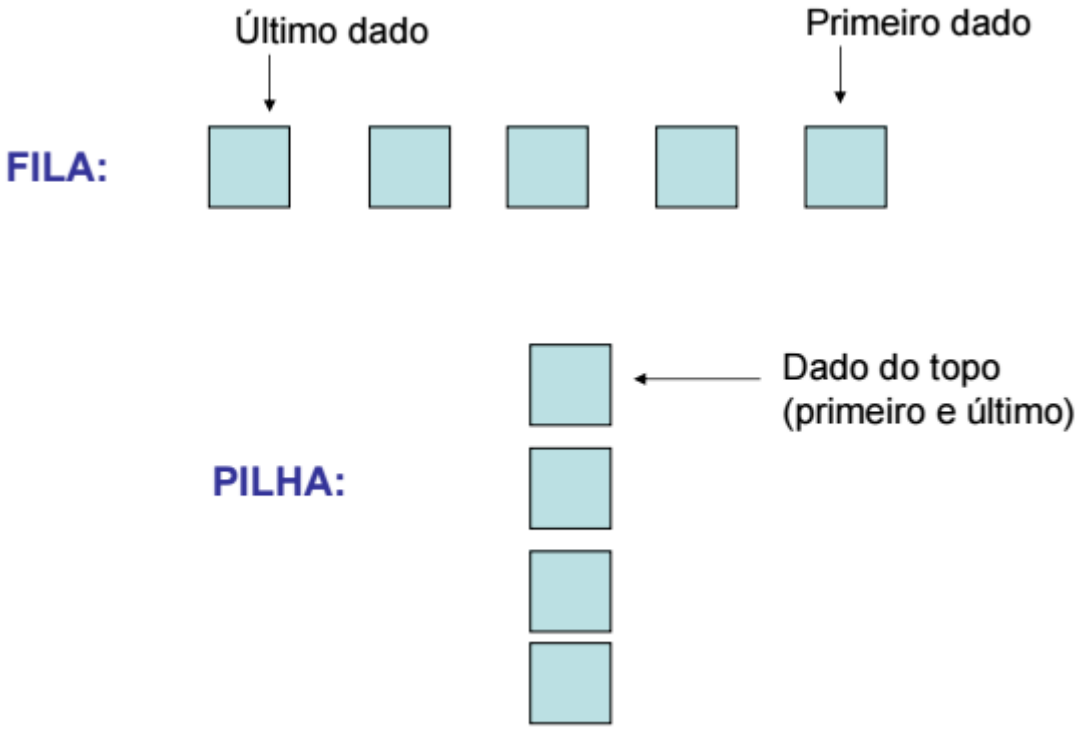
Um algoritmo é um **processo sistemático** para computar um resultado a partir **de dados de entrada**. Temos então que um algoritmo precisa tratar de **argumentos de forma lógica** a fim de que possamos tirar **conclusões**.

Então como vamos representar esse processo sistemático de maneira computacional? Precisamos criar alguma organização a fim de operar os nossos dados de entrada e saída, correto?

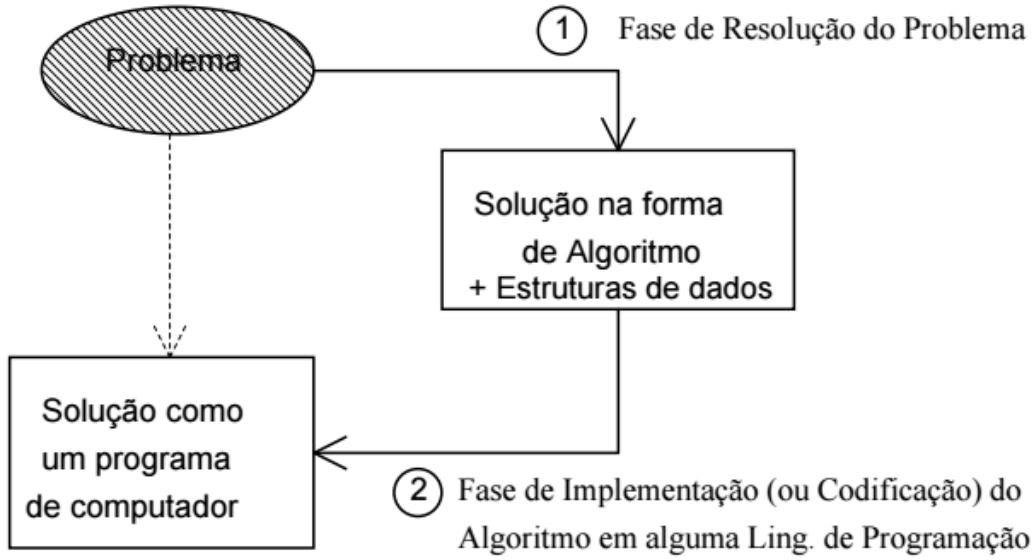
Nasce então o que conhecemos por Estrutura de Dados

Professor defina estrutura de dados? Estrutura de dados é a maneira pela qual **organizamos os dados a fim de operar sobre eles**.

Exemplo de duas estruturas de dados (Fila e Pilha):



Perceba então que ao reunir **estruturas de dados** (que organizam os dados) e algoritmos eu posso criar programas de computador. Logo: **estrutura de dados** + **algoritmos** = **programas**. Ok?



Quando desejamos projetar ou implementar uma estrutura de dados, precisamos de:

1) Uma **modelagem abstrata** dos **objetos** a serem manipulados e das **operações** sobre eles.

2) Uma **modelagem concreta** do tipos Abstratos de Dados (TAD – Abstract Data Type).

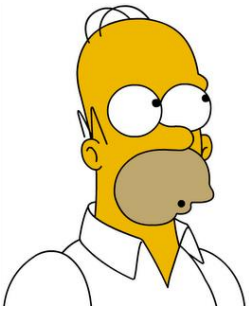
O que é um TAD? TAD (Tipos Abstratos de Dados) é um **modelo matemático que visa representar um conjunto de operações sobre um conjunto de valores**. Um TAD é normalmente representado através de um especificação algébrica que contém três partes: **Semântica, Sintática e Restrições**.

Temos então que a especificação **Semântica** trata **do comportamento de um tipo abstrato de dados**, enquanto o nível **sintático** define a **apresentação de um tipo abstrato de dados**, através da definição do nome do tipo, suas operações e seus argumentos.

As restrições como o próprio nome diz, **são responsáveis por estabelecer condições para a aplicação das operações**. Ok? Temos então que o TAD é responsável por **encapsular as estruturas de dados com características semelhantes**.

Podemos perceber que ao estudar estrutura de dados, nos preocupamos muito em como nossos dados **serão armazenados e manipulados** e isso passa diretamento pelo bendito "**dado**".

Temos então dois conceitos interessantes para trabalhar com dados dentro de estrutura de dados, temos os **dados homogêneos** e os **dados heterogêneos**.



Dados homogêneos são aqueles que possuem só **um tipo básico de dados** (Inteiros).

Dados heterogêneos são dados que possuem **mais de um tipo básico de dados** (Ex: Caracteres e Inteiros).

Os **vetores** por exemplo são estruturas de dados que trabalham somente com os **dados homogêneos**. Vetores representam estruturas nas quais todos os elementos são do mesmo tipo.

Existem também estruturas de dados que trabalham com **dados heterogêneos**, como as **Listas** que apresentam todos os seus elementos de tipos diferentes básicamente.

Temos também dois tipos de estruturas dentro do universo das estruturas de dados: **Estruturas Lineares** e **não Lineares**. As **Estruturas Lineares** são **estruturas em que cada elemento pode ter um único predecessor**, menos o primeiro elemento obviamente e **um único sucessor**, menos é claro o último elemento. São elas as Pilhas, Arranjos, Filas e Listas, dentre outros.

As **Estruturas não lineares** apresentam cada elemento **podendo ter mais de um predecessor ou sucessor**. Por exemplo: Os Grafos e as Árvores, dentre outras.

Dados Homogêneos: **Só um tipo básico de dados (**Inteiros**)**

Dados Heterogêneos: **São dados que possuem mais de um tipo básico de dados (**Caracteres e Inteiros**)**.

Estruturas Lineares: **Estruturas em que cada elemento pode ter um único predecessor (**Pilhas, Arranjos, Filas e Listas**)**.

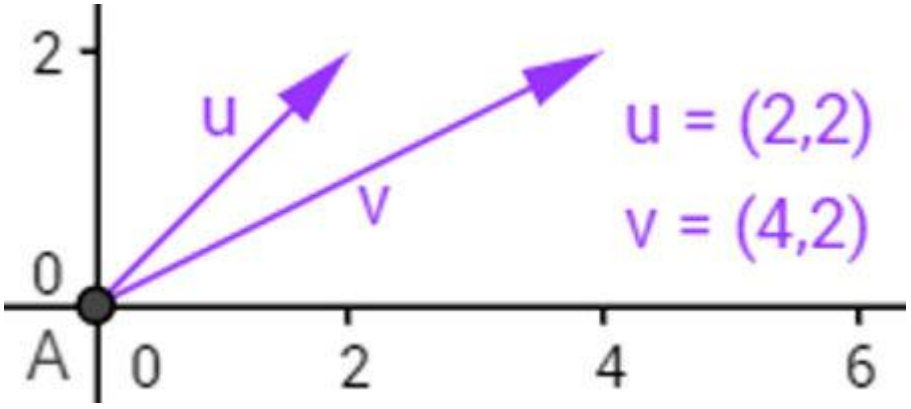
Estruturas não Lineares: Apresentam cada elemento **podendo ter mais de um predecessor ou sucessor (**Grafos e as Árvores**)**.

Perceba que quando tratamos de algum tipo de estrutura de dados, exemplo: "Listas", nós estamos tratando de um **tipo abstrato de dados** que tem comportamentos definidos. Ok?

Temos então **operações associadas a cada tipo de estrutura de dados**, e essas operações **são independentes de qualquer tipo de linguagem de programação ou paradigma**, logo não importa se implementamos uma Lista, Pilha ou Fila com paradigma estruturado ou com paradigma orientado a objeto.

2 – Vetores

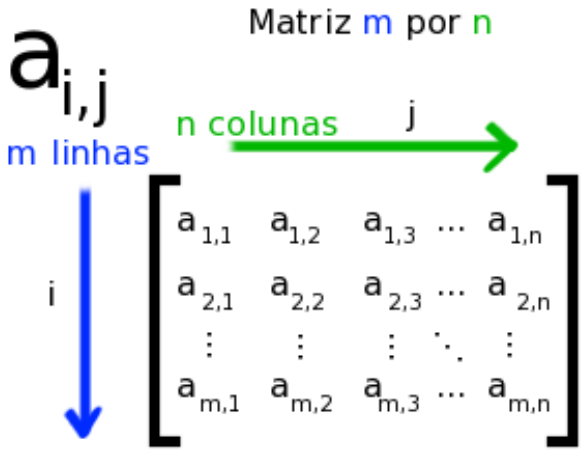
Vamos então compreender o que são essas estruturas de dados, definidas como **Matrizes e Vetores**. Primeiro o que é um Vetor?



Na matemática (essa linda ciência) um vetor é a representação através de setas, cujas coordenadas correspondem ao um ponto específico: u = (2,2) e v = (4,2), indicando também sentido, direção e intensidade. Ok?

Aqui no mundo computacional também temos vetores, eles são representados como **estruturas de dados homogêneas**, armazenando somente uma lista de valores do mesmo tipo, **podendo ser estático ou dinâmico**.

Professor e o que é uma matriz?

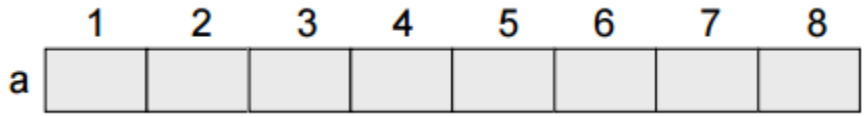


Matriz é um arranjo que visa representar um conjunto numérico, com linhas e colunas agrupadas (mxn). Logo temos na computação a representação de uma matriz como um **arranjo bidimensional ou multidimensional de alocação estática e sequencial**.

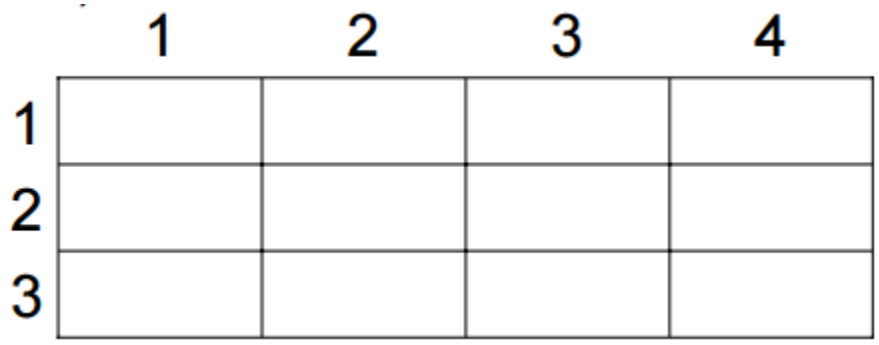
Em uma matriz todos os elementos são do mesmo tipo, com cada célula contendo somente um valor, os elementos dentro de uma matriz podem ser alocados linha por linha ou coluna por coluna.

Note que uma matriz pode ser considerada então um **vetor com mais de uma dimensão** com todos os elementos compondo o mesmo tipo de dados.

Vejamos o exemplo então de um vetor "a" com 8 posições: int Vetor [8];



Vejamos uma matriz com 3 linhas e 4 colunas: int Matriz [3][4];



3 – Lista encadeada linear

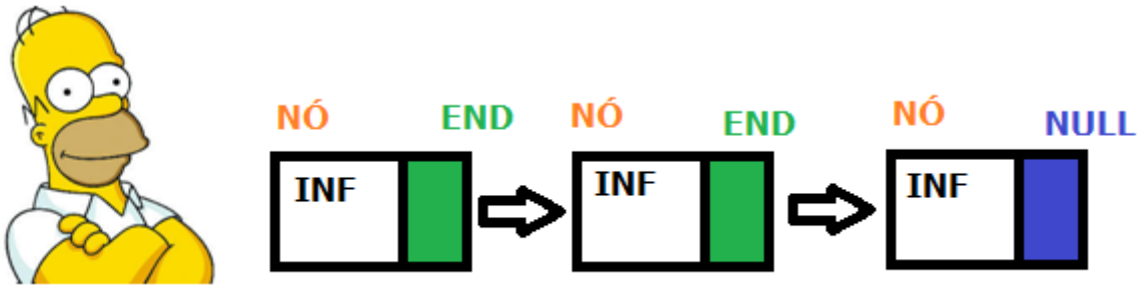
Vamos tratar agora sobre Lista Encadeada Linear ou apenas Lista Encadeada para os mais íntimos, que nada mais é que uma **estrutura de dados dinâmica** composta por uma **sequência de elementos encadeados** conhecidos como **nós**, possuindo dois campos: **campo endereço** e **campo informação**.

O **campo endereço** é utilizado para acessar o nó, que é o que formalmente conhecemos no universo computacional de forma didática como **ponteiro** ou **apontamento**. A lista então é acessada através do **ponteiro** que vai apontar para o **endereço do primeiro nó da lista** através de uma variável ou por referência.

Na lista o campo do endereço do **último nó conterá um valor NULL**, que nada mais é que um **endereço inválido**. Essa indicação é utilizada a fim de indicar o término ou final da lista.

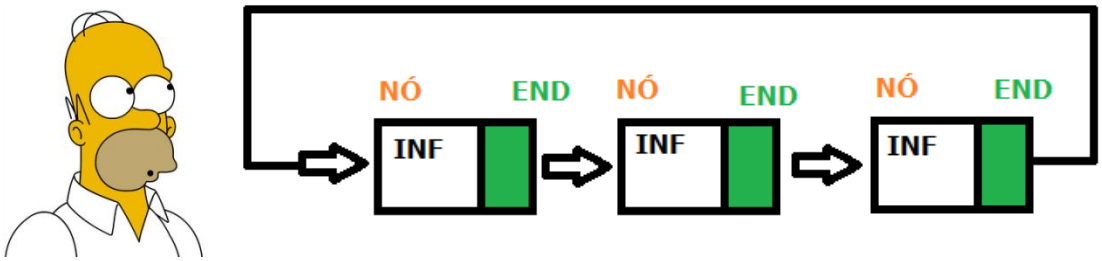
Uma lista pode ser inicializada como uma lista vazia e quando uma lista esta vazia, ela é uma lista nula, ou seja, **um lista nula é uma lista que não tem nós ou com apenas um nó**.

Exemplo:



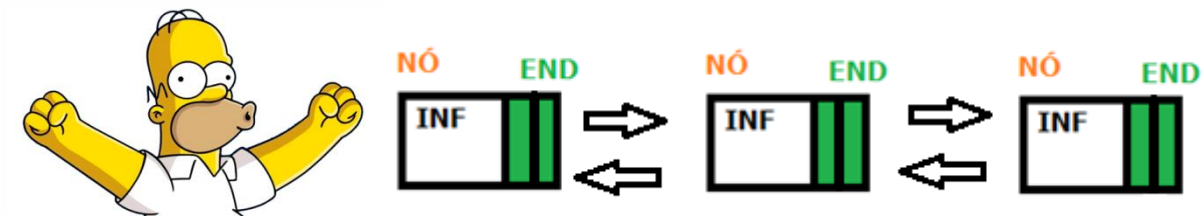
O último nó também é conhecido como **sentinela** e esta designação também pode ser utilizada para se referir ao primeiro elemento. Caso no último nó tenhamos um ponteiro apontando para o primeiro nó, teremos o que é conhecido como **Lista fechada** ou **Lista circular**.

Exemplo:



Uma lista circular **não apresenta nem o primeiro nem o último nó**, logo o ponteiro deverá apontar para o último nó arbitrariamente e o nó seguinte será estabelecido como o primeiro nó.

Existe também a **Lista duplamente encadeada**, que surgiu a fim de prover uma travessia no sentido contrário entre os nós, perceba que a lista encadeada linear e a lista circular permitem a travessia entre os nós em **apenas uma única direção**, para solucionar esse problema as lista duplamente encadeadas utilizam o apontamento do ponteiro **no primeiro elemento para o último elemento e o ponteiro seguinte do último para o primeiro elemento**. Vamos desenhar que fica mais simples:



Note que aqui cada nó terá **dois ponteiros**, um para o nó anterior e um para o nó posterior. As listas duplamente encadeadas podem ser chamadas de listas duplamente ligadas e podem ser lineares ou circulares.

Na lista duplamente encadeada os nós possuem três campos: **INF (Informação), Left** e **Right** que contêm ponteiros para os nós de ambos os lados. Perceba que o ponteiro do último elemento pode ser utilizado para percorrer a lista em **ordem inversa**.

Existem cinco operações que se aplicam a uma lista encadeada

Criação: A lista é criada em memória.

Busca: São pesquisados os nós na lista.

Inclusão: Novos nós são inseridos na lista.

Remoção: Nós existentes são retirados da lista.

Destruição: A lista é apagada.

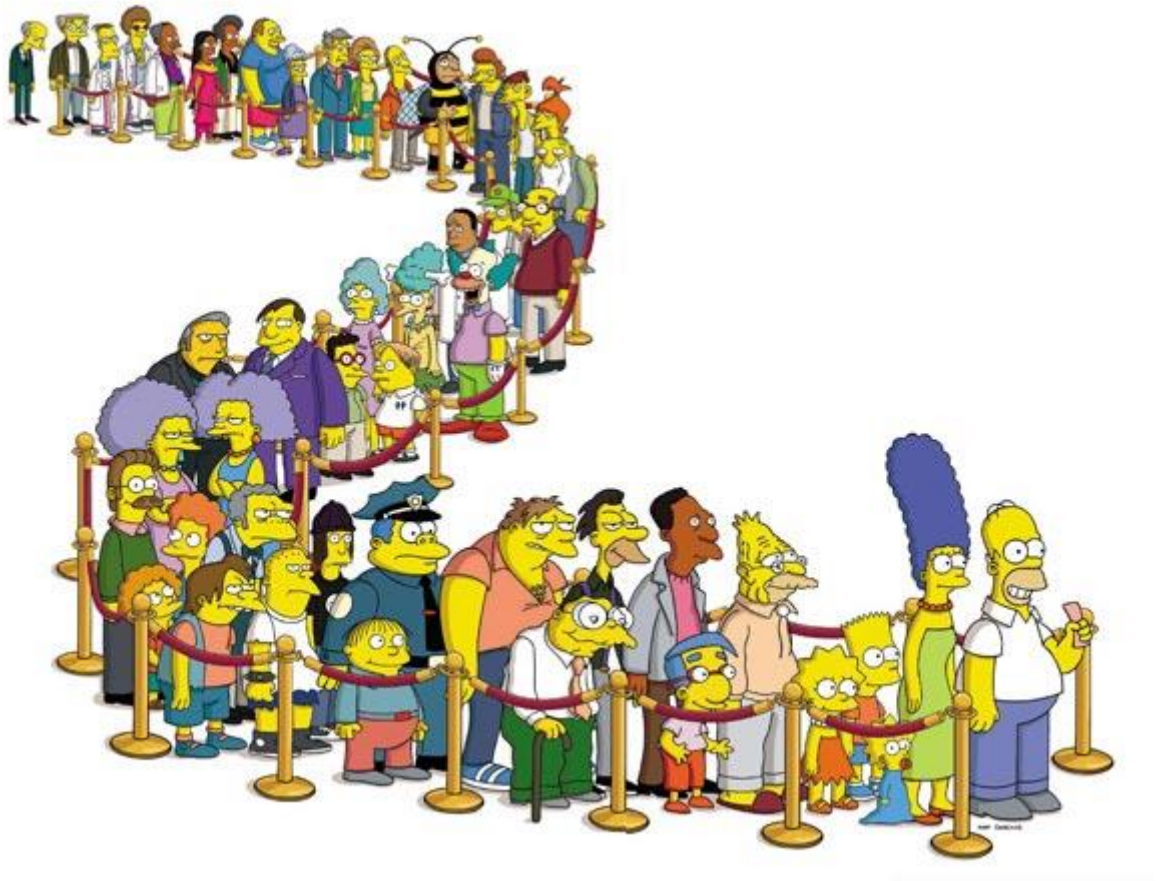
Antes da estrutura de dados em forma de lista os Sistemas Operacionais enfrentavam muitos problemas com fragmentação que nada mais é que desperdiçar espaço de memória entre alocações de elementos em memória.

Para resolver esse problema começamos a utilizar listas encadeadas porque elas eliminam o problema da fragmentação, tendo em vista que cada elemento é organizado de forma encadeada em memória, já que cada bloco tem um ponteiro para o próximo bloco.

Professor mais então Vetor e Lista são a mesma coisa? Qual a diferença entre um e o outro?

Uma lista percorre nó por nó a fim de acessar um dado. Note que comparando com a estrutura de dados "vetor" temos no vetor acesso direto ao dado, logo é possível acessar o dado sem percorrer nó por nó como na lista. Ok?

4 – Filas



Uma Fila é uma estrutura de **dados dinâmica e ordenada**, onde os elementos dentro da Fila seguem a seguinte regra: para sair da fila os elementos precisam estar em uma extremidade, chamada início da fila e para entrar na fila precisam estar na outra extremidade, chamada final da fila.

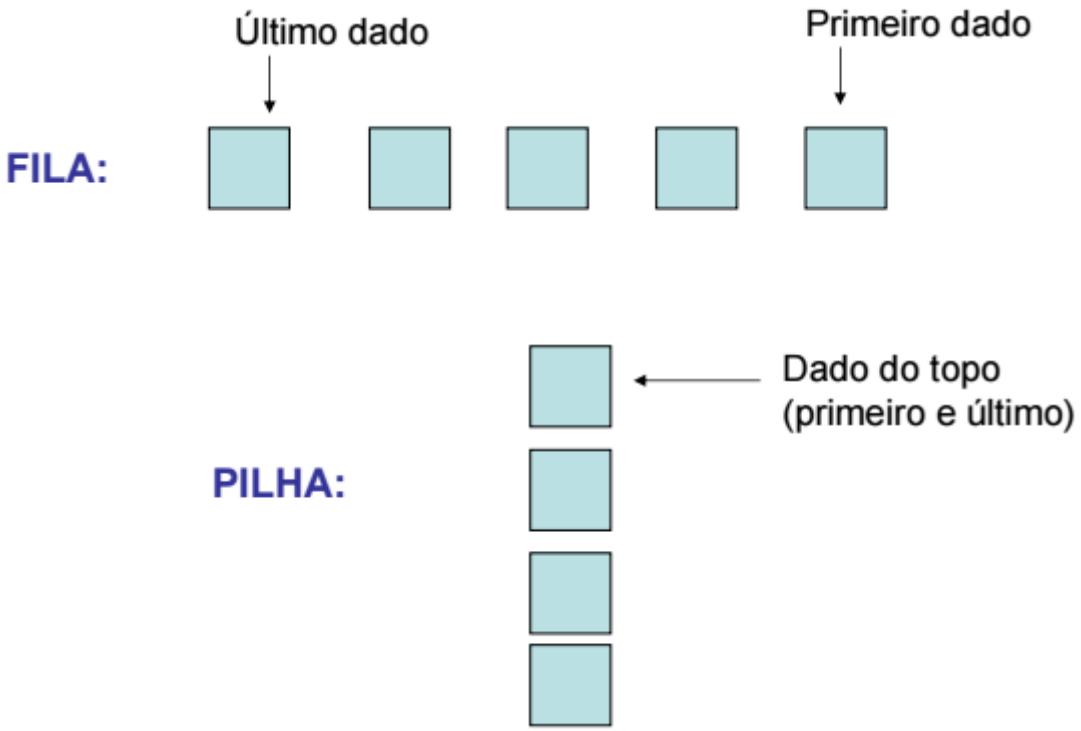
Só para ficar claro, quem está no início da fila é o Homer:  e quem esta no final da fila é o Burns:  Digo isso, porque é normal quando tratamos da estrutura de dados Fila, os alunos (as) confundirem o início com o final da fila. Ok?

A Fila também tem outro nome muito conhecido: **Lista FIFO (First In First Out)** que expressa a ideia por de trás da Fila. Temos então que na estrutura de dados Fila, o primeiro a entrar (**First In**) é o primeiro a sair (**First Out**).

Quando um elemento entra na fila, ele vai ocupar o final da fila, onde esta o "Burns", logo o elemento que esta mais perto de ser retirado da fila é o que está no início. Temos então duas operações dentro de uma Fila: Enfileirar e Desenfileirar, basicamente. Por fim também temos as **filas duplamente encadeadas** que permitem a saída e a entrada de elementos em uma fila pelas duas extremidades (início e final da fila).

5 – Pilhas

Já aprendemos o funcionamento da estrutura de dados Fila, então queria te recordar a primeira ilustração que fiz na aula para referenciar uma estrutura de dados, a fim de que possamos melhor compreender o que é uma pilha:

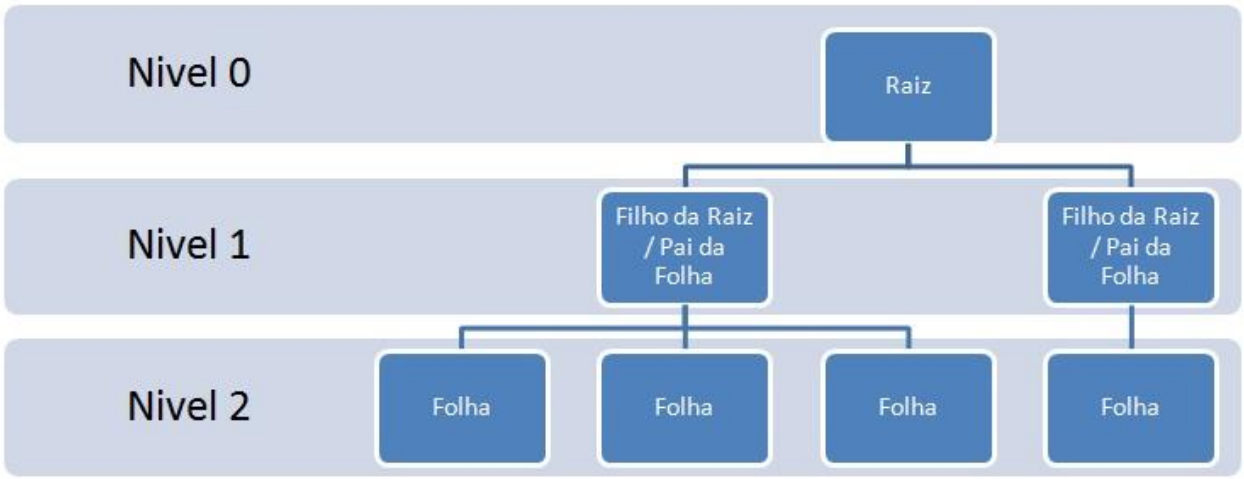


Na pilha nós temos um conjunto de elementos ordenados que podem sofrer inserção de elementos e retirada de elementos **apenas em uma extremidade**, que é o **topo da pilha**. Para implementar uma pilha podemos utilizar **listas ou vetores**.

Temos então aqui outro tipo de Lista, a lista **LIFO** (**Last In First Out**), ou seja, o último a entrar (**Last In**) será o primeiro a sair (**First Out**). Temos aqui três operações: **push**: responsável por inserir novos elementos no topo da pilha, **pop**: responsável por remover elementos do topo da pilha e **check** ou **top**: responsável por consultar o elemento no topo da pilha.

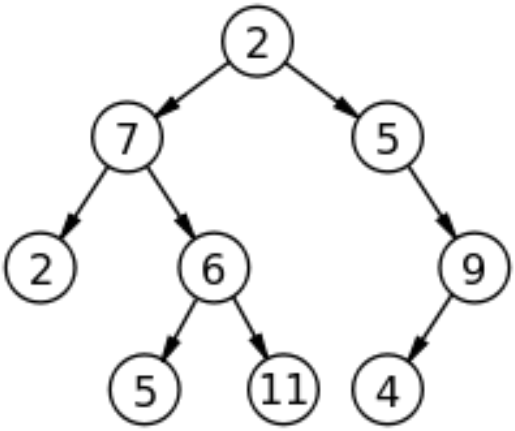
6 – Árvores

Árvores são estruturas de dados que trabalham com **hierarquia** de forma **não linear**, tendo um conjunto de **elementos finito** com apenas um **único elemento raiz**, com **sub-árvores** ligadas a esse elemento (raiz). Assim como temos um elemento raiz, temos nas extremidades da árvore "**nós**" que podem ser "**nós pais**" ou "**nós folhas**".



Temos aqui uma árvore com três níveis (0, 1 e 2) por padrão **a raiz representa o nível 0** e os outros níveis são subsequentes ao da raiz, formando assim nós pais e nós folha. A altura da árvore é igual ao número de níveis, no caso aqui temos uma árvores de altura 3. Ok?

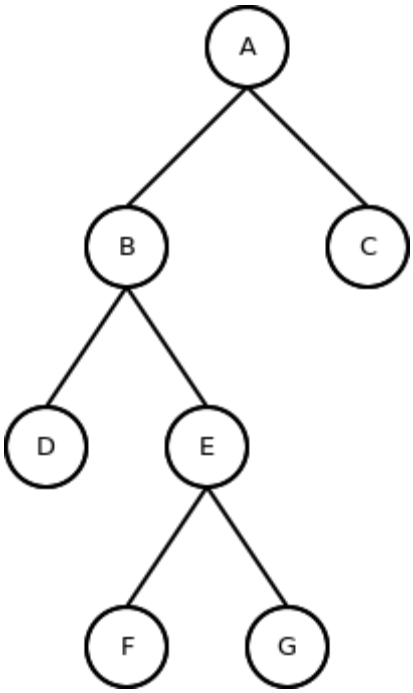
Existe uma classificação de árvore muito utilizada chamada de **árvore binária**, que é uma **árvore hierárquica em que todos os nós têm grau 0, 1 ou 2**.



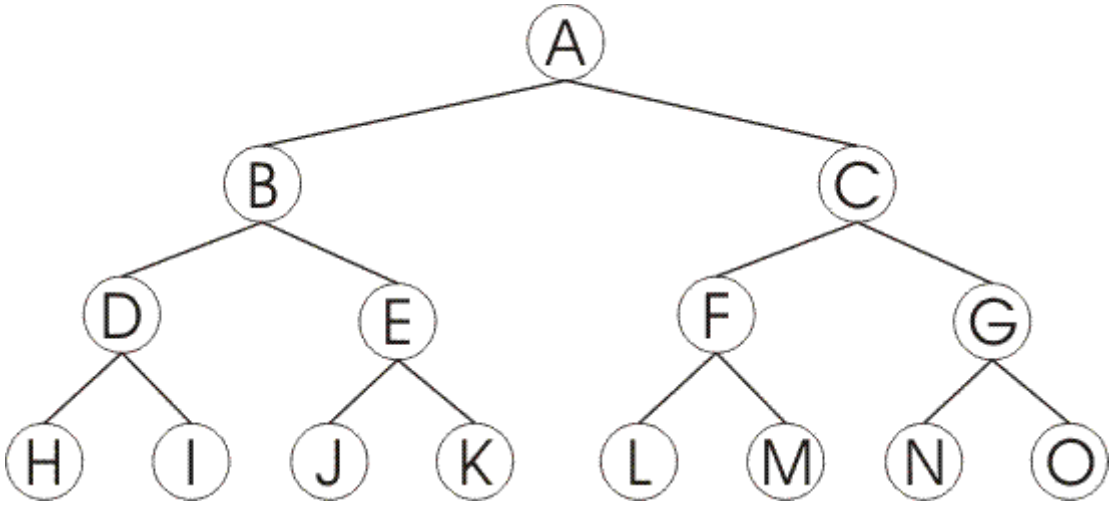
A árvore binária é muito utilizada na computação devido a sua característica de **recursividade**, note que cada nó poderá ter até duas folhas com ponteiros apontando dos "nós pai" para os "nós folha".

Temos também outros dois tipos de árvores parecidas com a árvore binária, que é a **árvore estritamente binária** que apresenta **todos os nós com grau 0 ou 2** e a **árvore binária completa** que apresenta **todas as folhas no mesmo nível.**

Árvore estritamente binária (Grau 0 ou 2)



Árvore binária completa (Todas as folhas no mesmo nível)



QUESTÕES COMENTADAS

1. **(FCC – Prefeitura de Teresina - 2016)** Considerando a estrutura de dados denominada árvore,

a) a sua altura é definida como a profundidade média de todos os seus vértices.
b) um vértice com um ou dois filhos é denominado folha.
c) cada nó tem no mínimo dois filhos em uma árvore binária.
d) as folhas de uma árvore binária completa podem ter profundidades distintas entre si.
e) a profundidade de um vértice em uma árvore é definida como o comprimento da raiz da árvore até esse vértice.

Comentários:

Galera, vamos passar alternativa por alternativa: (a) Errado! Altura é definida pela folha mais profunda; (b) Errado! Folhas não possuem filhos, do contrário não seriam folhas (as arves... somo nozes... péssimo, professor :P); (c) Errado! Os nós de uma árvore binaria podem ter NO MÁXIMO dois filhos, e não no mínimo; (d) Errado! Uma árvore binária completa é aquele em que todos os nós internos possuem seus dois filhos (máximo). Desse modo, todas as folhas devem ter a mesma profundidade; (e) Certo!

Gabarito: Letra E

---

2. **(CESPE - STM - 2011)** Enquanto uma lista encadeada somente pode ser percorrida de um único modo, uma árvore binária pode ser percorrida de muitas maneiras diferentes.

Comentários:

Galera, pense em uma árvore bem simples com um pai (raiz) e dois filhos. O Modo Pré-fixado vai ler primeiro a raíz, depois a sub-árvore da esquerda e depois a sub-árvore da direita. O Modo In-fixado vai ler primeiro a sub-árvore da esquerda, depois a raiz e depois a sub-árvore da direita. O Modo Pós-fixado vai ler primeiro a sub-árvore da esquerda, depois a sub-árvore da direita e depois a raiz.

Vamos resumir: o modo de percorrimento de uma árvore pode obedecer três regras de acordo com a posição da raiz (pai): pré-fixado (raiz, esquerda, direita); in-fixado (esquerda, raiz, direita); e pós-fixada (esquerda, direita, raiz).

Gabarito: CORRETO

---

3. **(FCC – TRT 9 - 2011)** Em uma árvore binária, todos os nós têm grau:

a) 2.
b) 0,1 ou 2.
c) divisível por 2.
d) maior ou igual a 2.
e) 0 ou 1.

Comentários:

Em uma árvore binária, todos os nós têm grau 0 (folha), 1 (único filho) ou 2 (dois filhos).

Gabarito: LETRA B

---

4. **(FCC – TRE-MT - 2010)** O uso de recursividade é totalmente inadequado na implementação de operações para manipular elementos de uma estrutura de dados do tipo árvore.

Comentários:

Pessoal, pelo contrário, é fundamental para implementação de operações.

Gabarito: ERRADO

---

5. **(FCC – MPE-AP - 2012)** A árvore é uma estrutura linear que permite representar uma relação de hierarquia. Ela possui um nó raiz e subárvores não vazias.

Comentários:

Pessoal, uma árvore é uma estrutura linear? Não, ela é uma estrutura hierárquica.

Gabarito: ERRADO

---

6. **(FCC -TRE-MT - 2010)** As listas, pilhas, filas e árvores são estruturas de dados que têm como principal característica a sequencialidade dos seus elementos.

Comentários:

Não! As árvores tem como principal característica a sequencialidade dos seus elementos.

Gabarito: ERRADO

---

7. **(FGV – DPE – MT - 2015)** No desenvolvimento de sistemas, a escolha de estruturas de dados em memória é especialmente relevante. Dentre outras classificações, é possível agrupar essas estruturas em lineares e não lineares, conforme a quantidade de sucessores e antecessores que

os elementos da estrutura possam ter. Assinale a opção que apresenta, respectivamente, estruturas de dados lineares e não lineares.

a) Tabela de dispersão
b) Estrutura de seleção e pilha
c) Pilha e estrutura de seleção
d) Pilha e árvore binária de busca
e) Fila e pilha

Comentários:

Pessoal, além dessa classificação, existe outra também importante: Estruturas Lineares e Estruturas Não-Lineares. As Estruturas Lineares são aquelas em que cada elemento pode ter um único predecessor (exceto o primeiro elemento) e um único sucessor (exceto o último elemento). Como exemplo, podemos citar Listas, Pilhas, Filas, Arranjos, entre outros. Já as Estruturas Não-Lineares são aquelas em que cada elemento pode ter mais de um predecessor e/ou mais de um sucessor. Como exemplo, podemos citar Árvores, Grafos e Tabelas de Dispersão. Essa é uma classificação muito importante e muito simples de entender. Pessoal, vocês perceberão que esse assunto é cobrado de maneira superficial na maioria das questões, mas algumas são nível doutorado!

Gabarito: Letra D

---

8. **(FUNCAB – SEJUS-RO - 2010)** Árvores são estruturas de dados estáticas com sua raiz representada no nível um.

Comentários:

Não! Árvores são estruturas dinâmicas e sua raiz, em geral, é representada no nível o (mas depende de autor para autor)

Gabarito: ERRADO

---

9. **(CESPE – DETRAN-ES - 2010)** Denomina-se árvore binária a que possui apenas dois nós.

Comentários:

Não, árvore binária é aquela em que cada nó tem, no máximo, dois filhos!

Gabarito: ERRADO

---

10. (CETAP – AL-RR - 2010) Uma árvore é composta por duas raízes, sendo uma principal e a outra secundária.

Comentários:

Não, uma árvore possui somente um nó raíz!

Gabarito: ERRADO

---

11. (CESPE - BANCO DA AMAZONIA - 2012) O tipo de dados árvore representa organizações hierárquicas entre dados.

Comentários:

Perfeito, observem que alguns autores tratam Tipos de Dados como sinônimo de Estruturas de Dados.

Gabarito: CORRETO

---

12. (CETAP- AL-RR - 2010) Uma árvore binária é aquela que tem como conteúdo somente valores binários

Comentários:

Não! Uma árvore binária é aquela que tem, no máximo, grau 2!

Gabarito: ERRADO

---

13. (CESPE - BANCO DA AMAZONIA - 2012) As operações de busca em uma árvore binária não a alteram, enquanto operações de inserção e remoção de nós provocam mudanças sistemáticas na árvore.

Comentários:

Perfeito! Operações de Busca não alteram nenhuma estrutura de dados. Já Operações de Inserção e Remoção podem provocar diversas mudanças estruturais.

Gabarito: CORRETO

---

14. (CESGRANRIO – TRANSPETRO - 2018) Analise a árvore binária de busca (BST), abaixo, representada pelas chaves dos seus nós.

Qual é a sequência de chaves representativa do seu percurso em pré-ordem?

a) 11; 12; 18; 23; 27; 45; 53; 71
b) 11; 18; 12; 27; 53; 71; 45; 23
c) 23; 12; 11; 18; 45; 27; 71; 53
d) 53; 71; 27; 45; 18; 11; 12; 23
e) 23; 45; 71; 27; 53; 18; 12; 11

Comentários:

Pessoal, Pré-Ordem: Lê o nó antes
In-Ordem (em Ordem): Lê o nó entre
Pós-Ordem: Lê o nó depois

A ordem de leitura das sub-árvores sempre é esquerda, depois direita. Em pré-ordem (nó antes), temos: NÓ-ESQUERDA-DIREITA. Portanto: 23-12-11-18-45-27-71-53

Gabarito: Letra C

---

15. (CESGRANRIO – TRANSPETRO - 2018) Considere uma árvore binária de busca (BST) com n (n>3) níveis (o nó raiz está no nível 1), 2n -1 nós e todas as chaves diferentes. Suponha, ainda, que algum dos pais de duas folhas seja removido da árvore e, mais tarde, uma chave com o mesmo valor da chave do nó removido seja inserida na árvore.

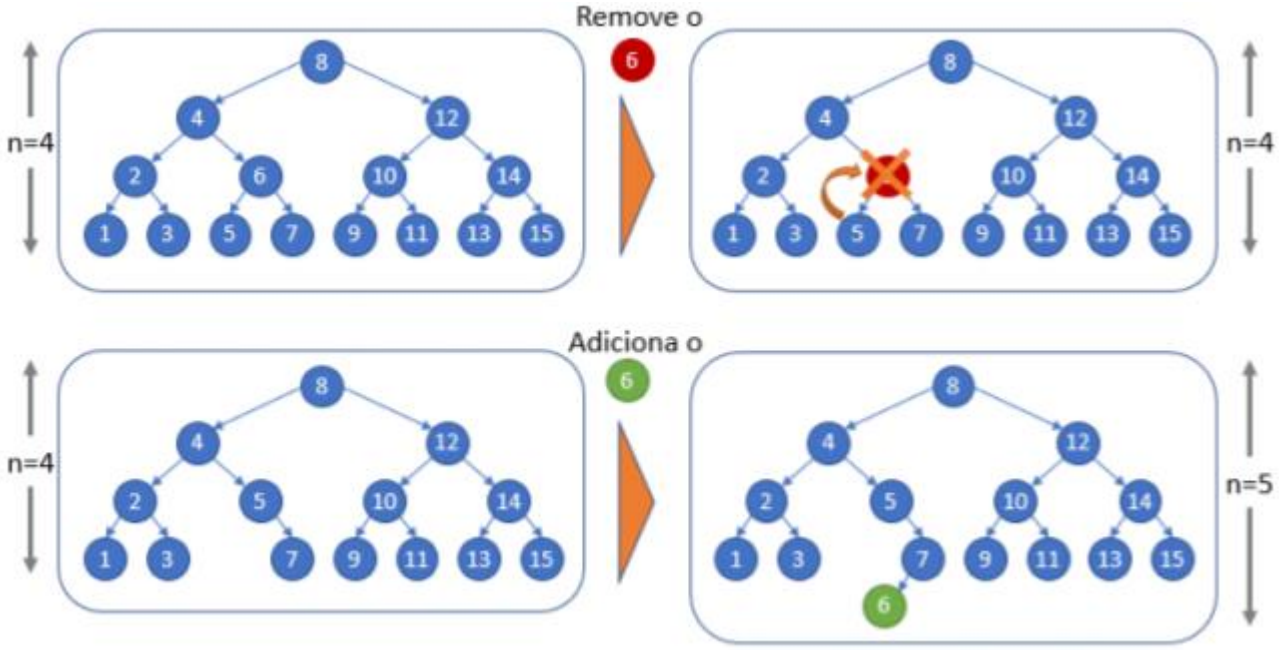
Quantas são as comparações necessárias para fazer a busca e encontrar o nó cuja chave foi removida e depois reinserida?

a) n – 2
b) n – 1
c) n
d) n +1
e) n + 2

Comentários:

Imaginando uma BST completa, quando removemos um nó pai, o menor valor da sub-árvore à direita assume o local do nó removido. Ao adicioná-lo novamente, esse valor será maior que o nó que assumiu sua posição, e seguirá para sub-árvore direita. Porém, o valor é menor do que o valor

do filho direito e, portanto, deverá se tornar filho esquerdo do filho direito, e a árvore ganhará 1 nível. Veja o exemplo abaixo, para n=4:



Logo, uma árvore de grau n, tornar-se-á n+1.

Gabarito: Letra D

---

16. (FCC – DPE-AM - 2018) Certo documento possui 1 milhão de palavras não repetidas e foi editado em um editor de textos. Considerando que o editor de textos utiliza uma Árvore Binária de Busca – ABB de altura mínima para armazenar as palavras digitadas de forma a facilitar sua localização, para se localizar qualquer palavra nesta estrutura de dados serão necessárias, no máximo,

a) 1 milhão de comparações.
b) 20 comparações.
c) 32 comparações.
d) log101000000 comparações.
e) 2 milhões de comparações.

Comentários:

1 milhão de palavras em uma árvore binária. Cada comparação em uma árvore binária diminui o universo pesquisado pela metade. Sendo assim, o número de comparações máximas sempre será log _{2} (N).

O valor exato de Log2(106) é 19.93, o seja, 19 testes não são o suficiente mas 20 certamente são. Uma aproximação que facilita muito vários cálculos é que 103 ≈ 2 10 (1.000 é ligeiramente menor que 1024).

Assim, 1 milhão é 106 , que é o mesmo que 103 * 103 , que será ligeiramente menor que 210*210 que é o mesmo que 2 20.

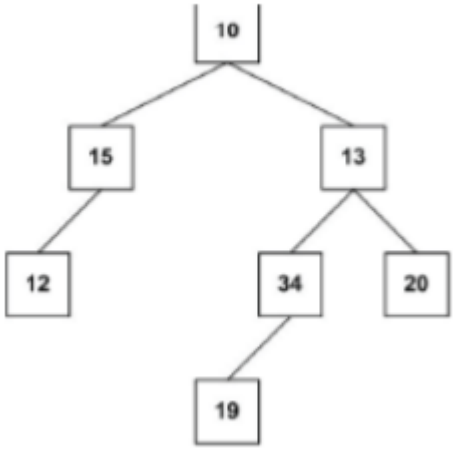
Ora, Log2(220) = 20, que será ligeiramente maior que Log2(106). Portanto, resposta correta Letra B, 20 comparações.

Gabarito: Letra B

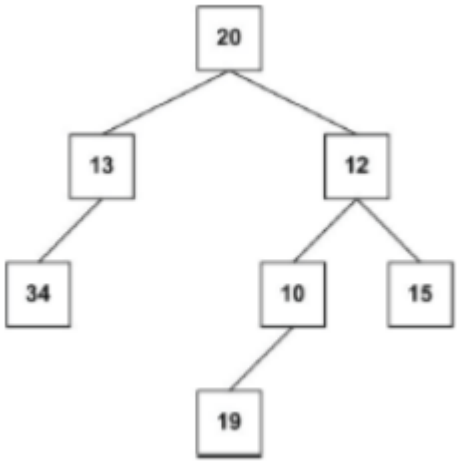
---

17. (CESGRANRIO – TRANSPETRO - 2018) Uma árvore binária foi percorrida em ordem simétrica, e os valores de seus nós exibidos no console. O resultado desse procedimento foi o seguinte: 15 12 10 19 20 13 34 Dentre as árvores apresentadas, a única capaz de produzir o resultado acima é:

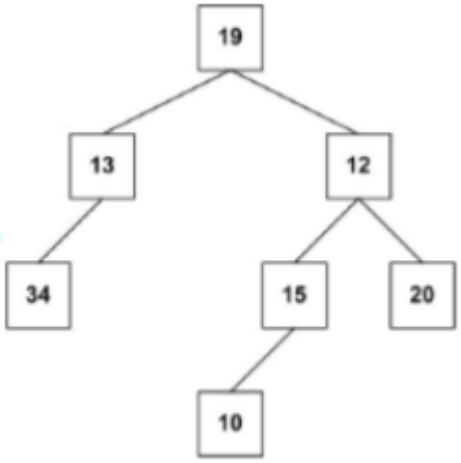
A)



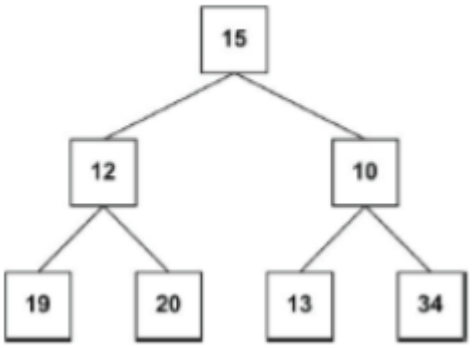
B)



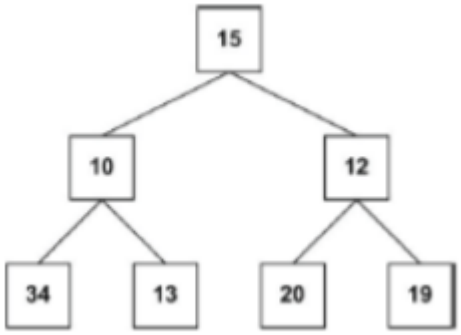
C)



D)



E)



Comentários:

Galera, a leitura em ordem é feita da seguinte forma: esquerda - nó - direita. O examinador inventou a leitura em ordem simétrica: direita - nó - esquerda. Curiosamente, não existe menção dessa forma de leitura em nenhuma literatura de referência... parece que foi coisa do examinador mesmo... Lendo as árvores seguindo: direita - nó - esquerda, temos:

a) 20-13-34-19-10-15-12

b) 15-12-10-19-20-13-34

c) 20-12-15-10-19-13-34
d) 34-10-13-15-20-12-19
e) 29-12-20-15-13-10-34

A ordem procurada é a da Letra B.

NOTA: infelizmente, a mesma banca já cobrou a "ordem simétrica" como esquerda-nó-direita... para mim, não existe de forma consolidada na literatura "em ordem simétrica" e, portanto, a questão deveria ter sido anulada.

Gabarito: Letra E

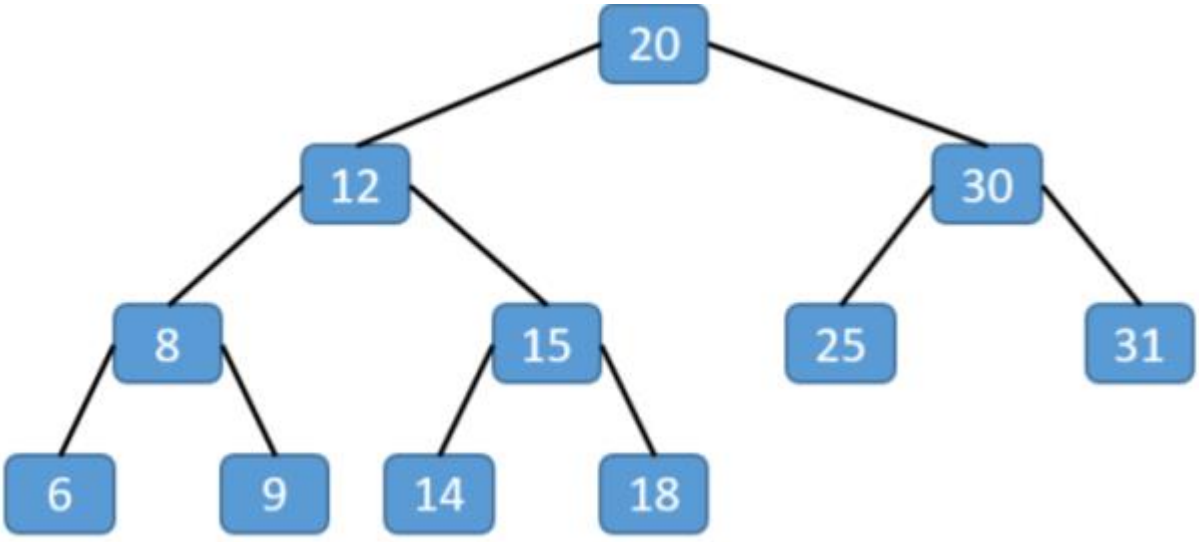
---

18. (CESGRANRIO – PETROBRÁS - 2018). A sequência de chaves 20 – 30 – 25 – 31 – 12 – 15 – 8 – 6 – 9 – 14 – 18 é organizada em uma árvore binária de busca. Em seguida, a árvore é percorrida em pré-ordem. Qual é a sequência de nós visitados?

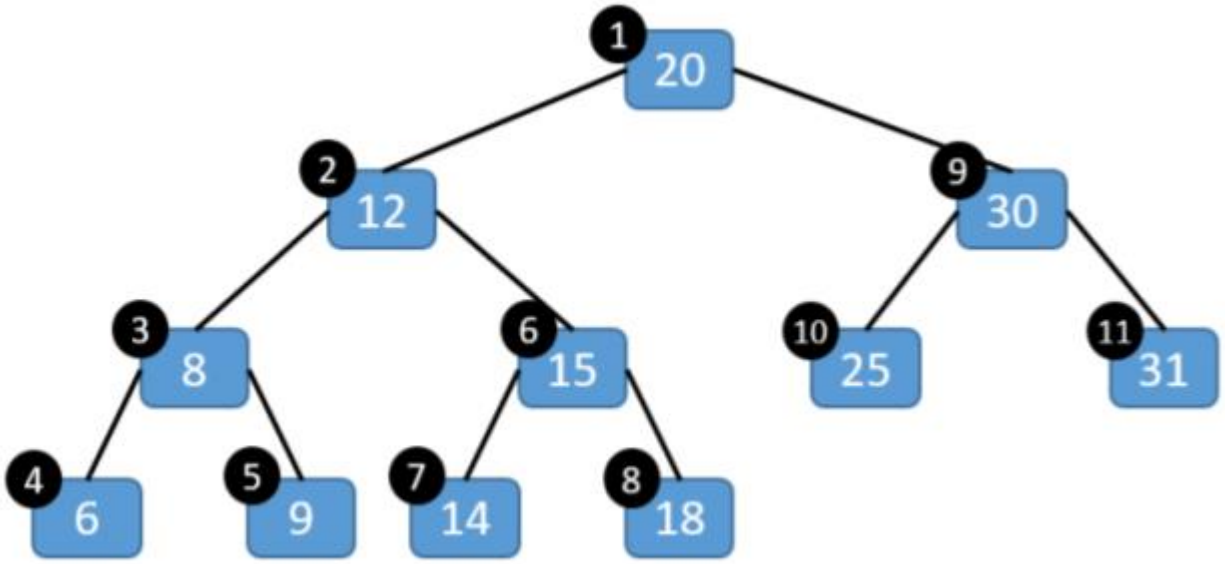
a) 6 – 9 – 8 – 14 – 18 – 15 – 12 – 25 – 31 – 30 – 20
b) 20 – 12 – 8 – 6 – 9 – 15 – 14 – 18 – 30 – 25 – 31
c) 6 – 8 – 9 – 12 – 14 – 15 – 18 – 20 – 25 – 30 – 31
d) 20 – 30 – 31 – 25 – 12 – 15 – 18 – 14 – 8 – 9 – 6
e) 6 – 8 – 9 – 14 – 15 – 18 – 12 – 25 – 30 – 31 – 20

Comentários:

A árvore binária de busca inclui os itens mantendo à esquerda os valores menores, e à direita os valores maiores. Sendo assim, a entrada 20 – 30 – 25 – 31 – 12 – 15 – 8 – 6 – 9 – 14 – 18 produzirá a árvore binária:



A leitura em Pré-Ordem lê os nós da árvore da seguinte forma: nó raiz, sub-árvore esquerda, sub-árvore direita. Para cada sub-árvore, repete-se: primeiro o nó, depois esquerda, depois direita. Sendo assim, os nós serão lidos na seguinte ordem (nr. de ordem no círculo preto):



Portanto: 20 – 12 – 8 – 6 – 9 – 15 – 14 – 18 – 30 – 25 – 31, Letra B.

Gabarito: Letra B

---

19. (CESGRANRIO – BASA - 2018) Uma árvore binária completa de busca, isto é, uma árvore em que todos os níveis têm o máximo número de elementos, tem um total de N nós. O número máximo de comparações necessárias para encontrar um elemento nessa árvore é

a) N
b) N 2
c) log 2(N+1)
d) (N+1)*log 2(N+1)
e) \sqrt{(N 1)}

Comentários:

Em uma árvore binária de busca, os elementos estão ordenados, sendo que os elementos menores que a chave do nó ficam na sub-árvore à esquerda, e os elementos maiores que a chave ficam na sub-árvore à direita. Dessa forma, a cada teste você tem:

• Se o valor procurado for igual à chave do nó, você encontrou o elemento;

• Se o valor procurado for menor que a chave do nó, procure na sub-árvore à esquerda;
• Se o valor procurado for maior que a chave do nó, procure na sub-árvore à direita.
• Como a árvore está completamente preenchida, a cada teste você irá encontrar o elemento, ou dividirá o conjunto de pesquisa na metade (sub-árvores à direita ou esquerda). Assim, você ira realizar, no máximo log2 (N +1) testes, Letra C.

Gabarito: Letra C

---

20. (CESPE – TER-PA - 2017) No estabelecimento de uma estrutura hierárquica, foi definida a seguinte árvore binária S:

S = (12(10(9(8))(11))(14(13)(15)))

Considerando o resultado da operação de exclusão do nó 12, assinale a opção que corresponde a nova estrutura da árvore S.

a) (10(9(8))(11(14(13)(15)))
b) (11(9(8)(10))(14(13)(15)))
c) (11(10(9(8))(14(13)(15)))
d) (13(10(9)(11))(14(15)))
e) (13(11(9)(10))(14(15)))

Comentários:

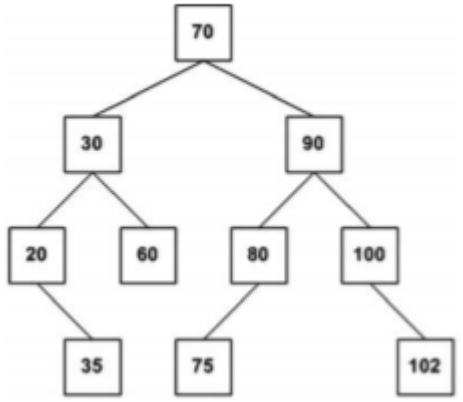
Conforme vimos em aula, a questão já inicia com um problema grave: ela não especifica qual tipo de árvore. Ainda assim, ela está errada e é fácil descobrir que não pode ser a Letra C (Gabarito Preliminar) porque a quantidade de parênteses abertos e fechados são diferentes – logo jamais poderia ser essa a resposta. Concluímos, então, que essa questão não possui resposta alguma, visto que falta fechar um parêntese. A Letra C (11(10(9(8))(14(13)(15))) está quase certa, mas faltou um parêntese: (11(10(9(8)))(14(13)(15))).

Gabarito: X

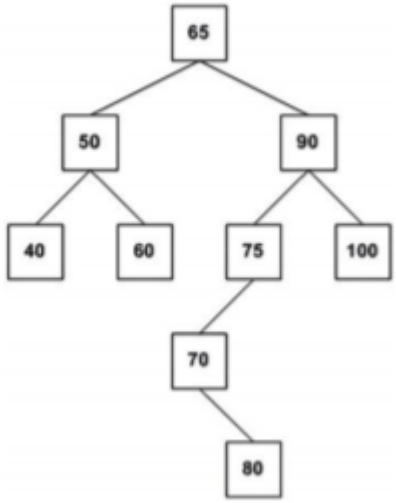
---

21. (CESGRANRIO – PETROBRÁS - 2012) Qual figura representa uma árvore AVL?

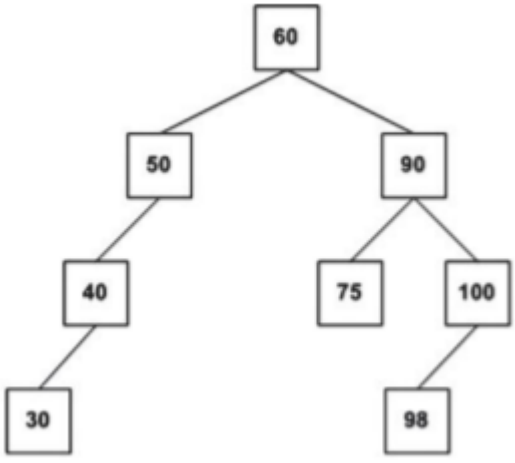
A)


70
30
90
20
60
80
100
35
75
102

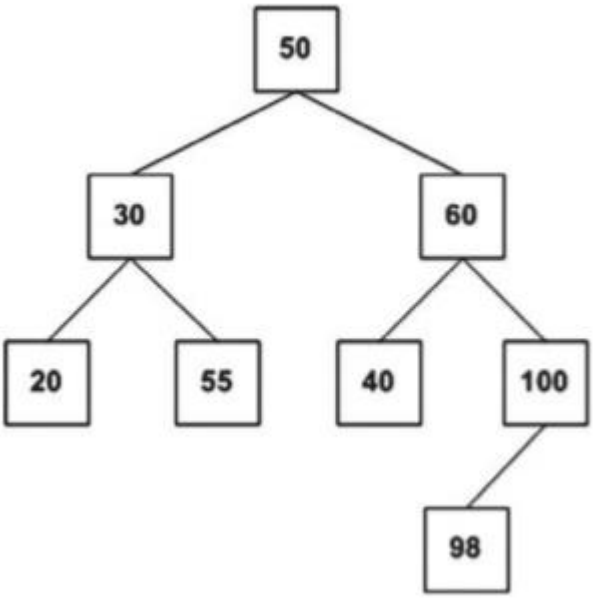
B)


65
50
90
40
60
75
100
70
80

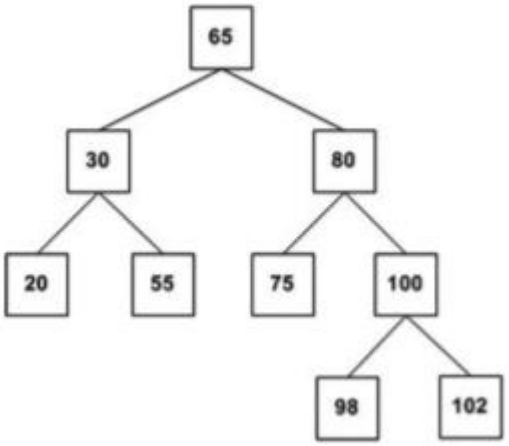
C)


60
50
90
40
75
100
30
98

D)



E)



Comentários:

Vamos relembrar dois conceitos: (1) Uma Árvore AVL é uma árvore binária de busca em que, para todos os seus nós, a diferença da altura da subárvore da esquerda para a altura da subárvore da direita deve ser no máximo 1. (2) Uma Árvore Binária de Busca é aquela em que, para todos os nós, a subárvore da esquerda possui um valor menor que a subárvore da direita. Dito isso, vamos analisar agora as opções:

(a) Errado. Não é uma árvore binária de busca, visto que o 35 é maior que 30 e está na subárvore da esquerda;

(b) Errado. Não é uma árvore binária de busca, visto que o 80 é maior que 75 e está na subárvore da esquerda;

(c) Errado. Observem que se trata realmente de uma árvore binária de busca, porém está desbalanceada. A diferença de altura da subárvore da esquerda de 50 e a subárvore da direita de 50 é 2 (que é maior do 1), portanto não é uma Árvore AVL.

(d) Errado. Não é uma árvore binária de busca, visto que o 55 é maior que 50 e está na subárvore da esquerda;

(e) Correto. Trata-se de árvore binária de busca e está completamente balanceada – requisitos para ser definida como uma Árvore AVL

Gabarito: LETRA E

---

22. (CESGRANRIO - PETROBRÁS - 2011) Após a inserção de um nó, é necessário verificar cada um dos nós ancestrais desse nó inserido, relativamente à consistência com as regras estruturais de uma árvore AVL.

PORQUE

O fator de balanceamento de cada nó, em uma árvore AVL, deve pertencer ao conjunto formado por {-2, -1, 0, +1, +2}. Analisando-se as afirmações acima, conclui-se que:

a) as duas afirmações são verdadeiras, e a segunda justifica a primeira.

b) as duas afirmações são verdadeiras, e a segunda não justifica a primeira.

c) a primeira afirmação é verdadeira, e a segunda é falsa.

d) a primeira afirmação é falsa, e a segunda é verdadeira.

e) as duas afirmações são falsas

Comentários:

Galera, a primeira frase está perfeita! Após inserir um novo nó, você tem que verificar os nós ancestrais para se certificar de que a Árvore AVL continua balanceada. No entanto, o fator de balanceamento de cada nó deve pertencer ao conjunto formado por {-1, 0, 1}. Lembrem-se: o módulo da diferença jamais pode ser maior do que 1, portanto a primeira está verdadeira e a segunda falsa.

Gabarito: LETRA C

---

23. (IBFC- TRE-AM - 2014) Quanto ao Algoritmo e estrutura de dados no caso de árvore AVL (ou árvore balanceada pela altura), analise as afirmativas abaixo, dê valores Verdadeiro (V) ou Falso (F) e assinale a alternativa que apresenta a sequência correta de cima para baixo:

( ) Uma árvore AVL é dita balanceada quando, para cada nó da árvore, a diferença entre as alturas das suas sub-árvores (direita e esquerda) não é maior do que um.

( ) Caso a árvore não esteja balanceada é necessário seu balanceamento através da rotação simples ou rotação dupla.

Assinale a alternativa correta:

a) F-F
b) F-V
c) V-F
d) V-V

Comentários:

A primeira alternativa está impecável, assim como a segunda. Vimos exaustivamente em aula!

Gabarito: LETRA D

---

24. (CESPE - PETROBRÁS - 2010) Uma sequência desordenada de números armazenada em um vetor é inserida em uma árvore AVL. Após a inserção nesta árvore, é feito um percurso em ordem simétrica (em ordem) e o valor de cada nó visitado é inserido em uma pilha. Depois de todos os nós serem visitados, todos os números são retirados da pilha e apresentados na tela. A lista de números apresentada na tela está:

a) ordenada ascendentemente de acordo com os números.
b) ordenada descendentemente de acordo com os números.
c) na mesma ordem do vetor original.
d) na ordem inversa do vetor original.
e) ordenada ascendentemente de acordo com sua altura na árvore.

Comentários:

Essa questão é legal! Olha só... Eu tenho um vetor com um bocado de valor desordenado. Esses valores são colocados em uma Árvore AVL. Ora, uma árvore AVL é uma árvore binária de busca, portanto segue suas propriedades. Logo, não importa se estava desordenado. À medida que são inseridos os valores desordenados na árvore, ela vai se balanceando e ordenando os dados. Após isso, vamos retirar os dados da Árvore AVL. Como? Da esquerda para a direita! E vamos colocá-los em uma pilha. Se estamos lendo da esquerda para a direita, estamos retirando do menor valor para o maior valor, logo o maior valor da pilha será o maior valor da Árvore AVL. Por

fim, ao retirar os elementos da pilha, retiramos do topo (maior) para a base (menor), logo em ordem descendente

Gabarito: LETRA B

---

25. (CESPE – PETROBRÁS - 2010) As árvores usadas como estruturas de pesquisa têm características especiais que garantem sua utilidade e propriedades como facilidade de acesso aos elementos procurados em cada instante. A esse respeito, considere as afirmações abaixo.

I - A árvore representada na figura (I) acima não é uma árvore AVL, pois as folhas não estão no mesmo nível.

II - A sequência 20, 30, 35, 34, 32, 33 representa um percurso sintaticamente correto de busca do elemento 33 em uma árvore binária de busca.

III - A árvore representada na figura (II) acima é uma árvore binária, apesar da raiz não ter filhos.

É (São) correta(s) APENAS a(s) afirmativa(s):

a) I.
b) II.
c) III.
d) I e II.
e) II e III.

Comentários:

(I) Errado. Trata-se, sim, de uma Árvore AVL; (II) Correto. Temos a sequência 20, 30, 35, 34, 32, 33 e estamos procurando o 33. 33>20, logo descemos à direita; 33>30, logo descemos à direita de novo; 3332, logo descemos à direita. Pronto, encontramos o 33; (III) Correto. Trata-se de um Árvore Binária sem filhos.

Gabarito: Letra E

LISTA DE QUESTÕES

1. **(FCC – Prefeitura de Teresina - 2016)** Considerando a estrutura de dados denominada árvore,

a) a sua altura é definida como a profundidade média de todos os seus vértices.
b) um vértice com um ou dois filhos é denominado folha.
c) cada nó tem no mínimo dois filhos em uma árvore binária.
d) as folhas de uma árvore binária completa podem ter profundidades distintas entre si.
e) a profundidade de um vértice em uma árvore é definida como o comprimento da raiz da árvore até esse vértice.

2. **(CESPE - STM - 2011)** Enquanto uma lista encadeada somente pode ser percorrida de um único modo, uma árvore binária pode ser percorrida de muitas maneiras diferentes.

3. **(FCC – TRT 9 - 2011)** Em uma árvore binária, todos os nós têm grau:

a) 2.
b) 0,1 ou 2.
c) divisível por 2.
d) maior ou igual a 2.
e) 0 ou 1.

4. **(FCC – TRE-MT - 2010)** O uso de recursividade é totalmente inadequado na implementação de operações para manipular elementos de uma estrutura de dados do tipo árvore.

5. **(FCC – MPE-AP - 2012)** A árvore é uma estrutura linear que permite representar uma relação de hierarquia. Ela possui um nó raiz e subárvores não vazias.

6. **(FCC -TRE-MT - 2010)** As listas, pilhas, filas e árvores são estruturas de dados que têm como principal característica a sequencialidade dos seus elementos.

7. **(FGV – DPE – MT - 2015)** No desenvolvimento de sistemas, a escolha de estruturas de dados em memória é especialmente relevante. Dentre outras classificações, é possível agrupar essas

estruturas em lineares e não lineares, conforme a quantidade de sucessores e antecessores que os elementos da estrutura possam ter. Assinale a opção que apresenta, respectivamente, estruturas de dados lineares e não lineares.

a) Tabela de dispersão
b) Estrutura de seleção e pilha
c) Pilha e estrutura de seleção
d) Pilha e árvore binária de busca
e) Fila e pilha

8. **(FUNCAB – SEJUS-RO - 2010)** Árvores são estruturas de dados estáticas com sua raiz representada no nível um.

9. **(CESPE – DETRAN-ES - 2010)** Denomina-se árvore binária a que possui apenas dois nós.

10. **(CETAP – AL-RR - 2010)** Uma árvore é composta por duas raízes, sendo uma principal e a outra secundária.

11. (CESPE - BANCO DA AMAZONIA - 2012) O tipo de dados árvore representa organizações hierárquicas entre dados.

12. (CETAP- AL-RR - 2010) Uma árvore binária é aquela que tem como conteúdo somente valores binários

13. (CESPE - BANCO DA AMAZONIA - 2012) As operações de busca em uma árvore binária não a alteram, enquanto operações de inserção e remoção de nós provocam mudanças sistemáticas na árvore.

14. (CESGRANRIO – TRANSPETRO - 2018) Analise a árvore binária de busca (BST), abaixo, representada pelas chaves dos seus nós.

Qual é a sequência de chaves representativa do seu percurso em pré-ordem?

a) 11; 12; 18; 23; 27; 45; 53; 71
b) 11; 18; 12; 27; 53; 71; 45; 23
c) 23; 12; 11; 18; 45; 27; 71; 53
d) 53; 71; 27; 45; 18; 11; 12; 23
e) 23; 45; 71; 27; 53; 18; 12; 11

15. (CESGRANRIO – TRANSPETRO - 2018) Considere uma árvore binária de busca (BST) com n (n>3) níveis (o nó raiz está no nível 1), 2n -1 nós e todas as chaves diferentes. Suponha, ainda, que algum dos pais de duas folhas seja removido da árvore e, mais tarde, uma chave com o mesmo valor da chave do nó removido seja inserida na árvore.

Quantas são as comparações necessárias para fazer a busca e encontrar o nó cuja chave foi removida e depois reinserida?

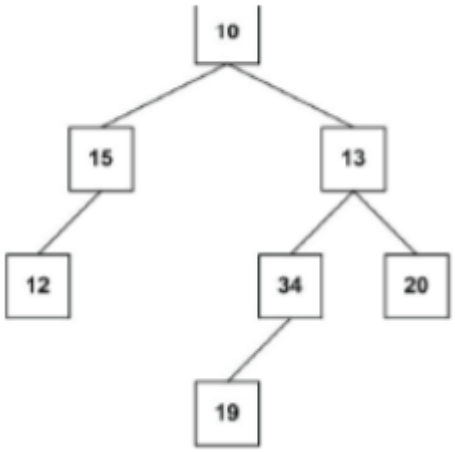
a) n – 2
b) n – 1
c) n
d) n +1
e) n + 2

16. (FCC – DPE-AM - 2018) Certo documento possui 1 milhão de palavras não repetidas e foi editado em um editor de textos. Considerando que o editor de textos utiliza uma Árvore Binária de Busca – ABB de altura mínima para armazenar as palavras digitadas de forma a facilitar sua localização, para se localizar qualquer palavra nesta estrutura de dados serão necessárias, no máximo,

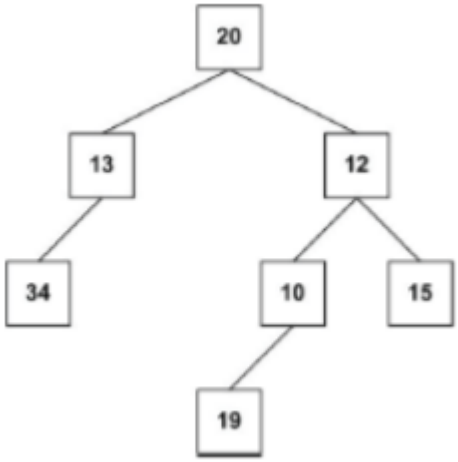
a) 1 milhão de comparações.
b) 20 comparações.
c) 32 comparações.
d) log101000000 comparações.
e) 2 milhões de comparações.

17. (CESGRANRIO – TRANSPETRO - 2018) Uma árvore binária foi percorrida em ordem simétrica, e os valores de seus nós exibidos no console. O resultado desse procedimento foi o seguinte: 15 12 10 19 20 13 34 Dentre as árvores apresentadas, a única capaz de produzir o resultado acima é:

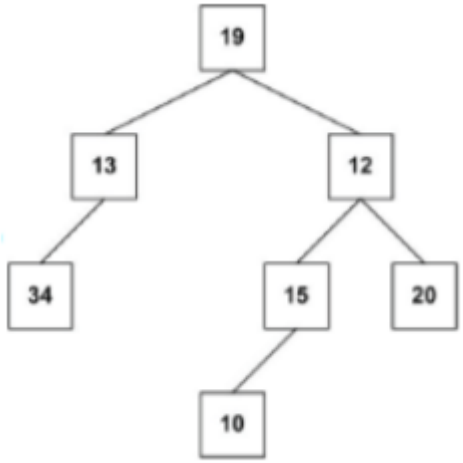
A)



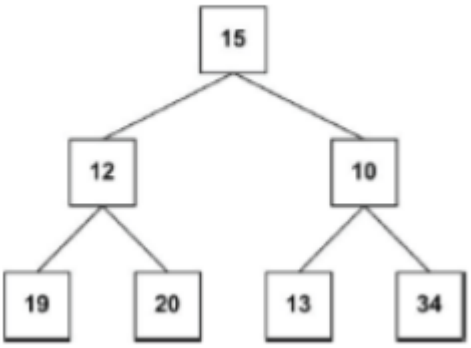
B)



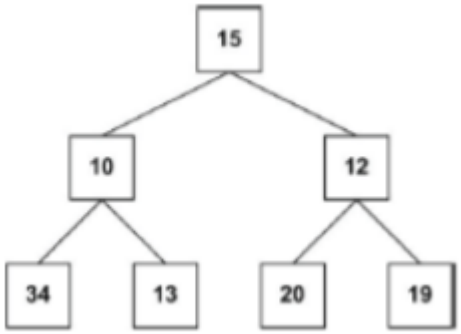
C)



D)



E)



18. (CESGRANRIO – PETROBRÁS - 2018). A sequência de chaves 20 – 30 – 25 – 31 – 12 – 15 – 8 – 6 – 9 – 14 – 18 é organizada em uma árvore binária de busca. Em seguida, a árvore é percorrida em pré-ordem. Qual é a sequência de nós visitados?

a) 6 – 9 – 8 – 14 – 18 – 15 – 12 – 25 – 31 – 30 – 20
b) 20 – 12 – 8 – 6 – 9 – 15 – 14 – 18 – 30 – 25 – 31
c) 6 – 8 – 9 – 12 – 14 – 15 – 18 – 20 – 25 – 30 – 31
d) 20 – 30 – 31 – 25 – 12 – 15 – 18 – 14 – 8 – 9 – 6
e) 6 – 8 – 9 – 14 – 15 – 18 – 12 – 25 – 30 – 31 – 20

19. (CESGRANRIO – BASA - 2018) Uma árvore binária completa de busca, isto é, uma árvore em que todos os níveis têm o máximo número de elementos, tem um total de N nós. O número máximo de comparações necessárias para encontrar um elemento nessa árvore é

a) N
b) N 2
c) log 2(N+1)
d) (N+1)*log 2(N+1)
e) \sqrt{(N 1)}

20. (CESPE – TER-PA - 2017) No estabelecimento de uma estrutura hierárquica, foi definida a seguinte árvore binária S:

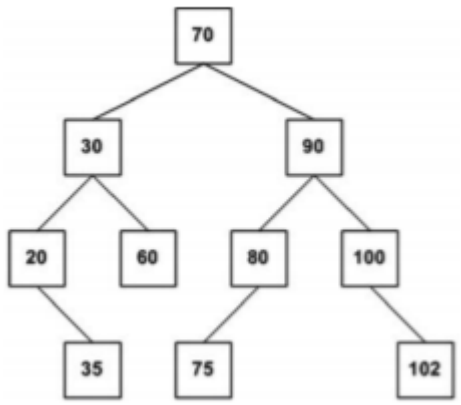
S = (12(10(9(8))(11))(14(13)(15)))

Considerando o resultado da operação de exclusão do nó 12, assinale a opção que corresponde a nova estrutura da árvore S.

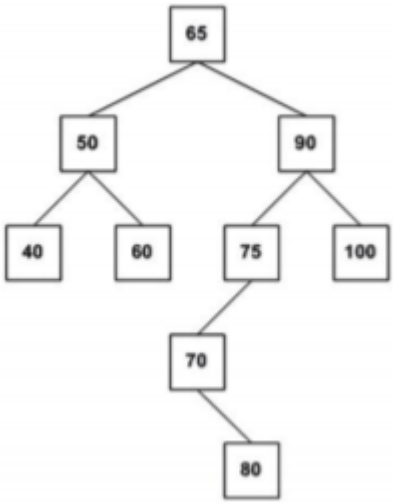
a) (10(9(8))(11(14(13)(15)))
b) (11(9(8)(10))(14(13)(15)))
c) (11(10(9(8))(14(13)(15)))
d) (13(10(9)(11))(14(15)))
e) (13(11(9)(10))(14(15)))

21. (CESGRANRIO – PETROBRÁS - 2012) Qual figura representa uma árvore AVL?

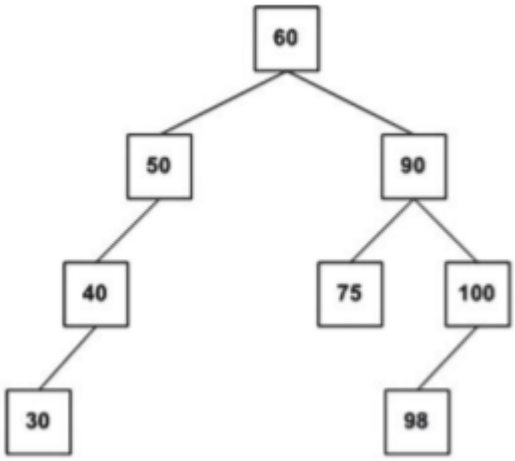
A)



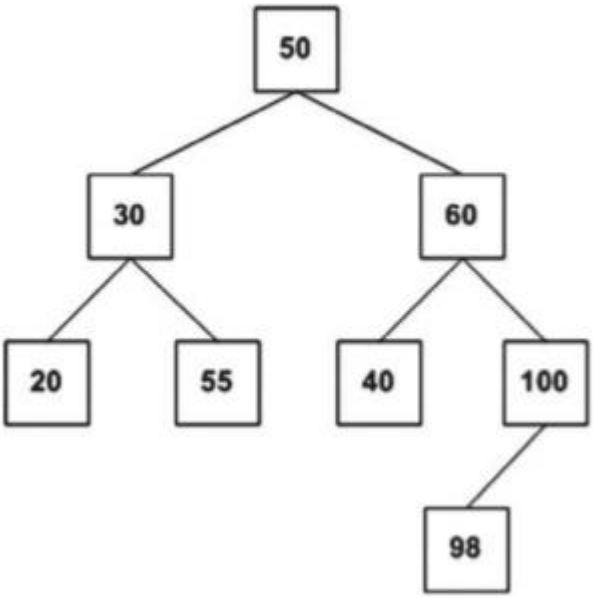
B)



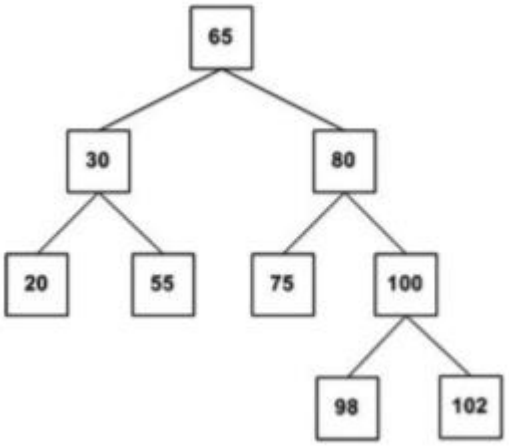
C)



D)



E)



22. (CESGRANRIO - PETROBRÁS - 2011) Após a inserção de um nó, é necessário verificar cada um dos nós ancestrais desse nó inserido, relativamente à consistência com as regras estruturais de uma árvore AVL.

PORQUE

O fator de balanceamento de cada nó, em uma árvore AVL, deve pertencer ao conjunto formado por {-2, -1, 0, +1, +2}. Analisando-se as afirmações acima, conclui-se que:

a) as duas afirmações são verdadeiras, e a segunda justifica a primeira.

b) as duas afirmações são verdadeiras, e a segunda não justifica a primeira.

c) a primeira afirmação é verdadeira, e a segunda é falsa.

d) a primeira afirmação é falsa, e a segunda é verdadeira.

e) as duas afirmações são falsas

23. (IBFC- TRE-AM - 2014) Quanto ao Algoritmo e estrutura de dados no caso de árvore AVL (ou árvore balanceada pela altura), analise as afirmativas abaixo, dê valores Verdadeiro (V) ou Falso (F) e assinale a alternativa que apresenta a sequência correta de cima para baixo:

( ) Uma árvore AVL é dita balanceada quando, para cada nó da árvore, a diferença entre as alturas das suas sub-árvores (direita e esquerda) não é maior do que um.

( ) Caso a árvore não esteja balanceada é necessário seu balanceamento através da rotação simples ou rotação dupla.

Assinale a alternativa correta:

a) F-F
b) F-V
c) V-F
d) V-V

:

24. (CESPE - PETROBRÁS - 2010) Uma sequência desordenada de números armazenada em um vetor é inserida em uma árvore AVL. Após a inserção nesta árvore, é feito um percurso em ordem simétrica (em ordem) e o valor de cada nó visitado é inserido em uma pilha. Depois de todos os nós serem visitados, todos os números são retirados da pilha e apresentados na tela. A lista de números apresentada na tela está:

a) ordenada ascendentemente de acordo com os números.
b) ordenada descendentemente de acordo com os números.
c) na mesma ordem do vetor original.
d) na ordem inversa do vetor original.
e) ordenada ascendentemente de acordo com sua altura na árvore.

25. (CESPE – PETROBRÁS - 2010) As árvores usadas como estruturas de pesquisa têm características especiais que garantem sua utilidade e propriedades como

facilidade de acesso aos elementos procurados em cada instante. A esse respeito, considere as afirmações abaixo.

I - A árvore representada na figura (I) acima não é uma árvore AVL, pois as folhas não estão no mesmo nível.

II - A sequência 20, 30, 35, 34, 32, 33 representa um percurso sintaticamente correto de busca do elemento 33 em uma árvore binária de busca.

III - A árvore representada na figura (II) acima é uma árvore binária, apesar da raiz não ter filhos.

É (São) correta(s) APENAS a(s) afirmativa(s):

a) I.
b) II.
c) III.
d) I e II.
e) II e III.

GABARITO

1. E
2. CORRETO
3. B
4. ERRADO
5. ERRADO
6. ERRADO
7. D
8. ERRADO
9. ERRADO
10. ERRADO
11. CORRETO
12. ERRADO
13. CORRETO
14. C
15. D
16. B
17. E
18. B
19. C
20. X
21. E
22. C
23. D
24. B
25. E

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.